Universidad Carlos III de Madrid
Ingeniería Técnica en Informática de Gestión

Desarrollo ágil de software a medida para PYMES

Leganés, octubre 2015

Proyecto fin de carrera de:
Antonio Mendoza Rodríguez

Proyecto dirigido por:
Gonzalo Génova Fuster

A mi madre.

Sin ella nada de esto habría sido posible.

Índice de contenido

1 Introducción

1.1 Propósito del documento

Este documento es una exposición técnica de un proyecto de software enfocado a ser una base para el desarrollo ágil de herramientas de gestión empresarial y fácilmente personalizables para empresas de diferentes sectores.

En el documento se estudiarán los objetivos a cumplir, se explicará en detalle cada uno de los componentes que forman parte del mismo para demostrar su potencial, justificar su conveniencia y convencer al lector de su rentabilidad de cara a su explotación comercial.

El documento abarcará principalmente el proyecto de diseño y desarrollo del software para terminar con un ejemplo práctico de su adaptación a una empresa real.

1.2 Visión general del documento

1.2.1 Descripción general

- Se presenta el proyecto y sus objetivos.
- Se determina el cliente objetivo y la idea de negocio para la explotación del producto.
- Se plantean las necesidades que pretende solucionar nuestro producto.

1.2.2 Requisitos

Se plantean una serie de requisitos funcionales y no funcionales genéricos y comunes a cualquier tipo de empresa que deben ser resolubles con nuestro producto.

1.2.3 Arquitectura

- Se exponen el modelo de datos utilizados en nuestro sistema.
- Se identifican los componentes del sistema.
- Se plantea la forma de resolver los requisitos planteados con el modelo de datos planteado.
- Se expone el funcionamiento del sistema.

1.2.4 Listados y tablas

Se explican algunos conceptos básicos para la comprensión de este documento.

1.2.5 Anexos.

- Guía de usuario.
- Guía de desarrollo.

1.3 Referencias

1.3.1 Software usados como referencia

- SysAid (Gestor de tareas): www.sysaid.com
- Golden .NET (Gestión contable): www.goldensoft.com
- OpenBravo (Plataforma ERP): www.openbravo.com

1.3.2 Guías y manuales

- http://www.w3schools.com/
- http://www.desarrolloweb.com/
- http://stackoverflow.com/
- https://developers.google.com
- http://www.wikipedia.org

1.3.3 Software y librearías de terceros

- PHP (Lenguaje de programación orientado a web): http://php.net/
- jQuery (librería JavaScript): http://jquery.com
- DataTables (librería JavaScript): http://datatables.net/
- Apache (Servidor HTTP): https://apache.org/
- Oracle MySQL (Sistema gestor de base de datos): https://www.mysql.com/

2 Descripción general

2.1 Perspectiva del producto

El producto final que pretende este proyecto es el desarrollo de un software para la gestión de empresas genérico, es decir, no enfocado a ningún cliente final pero que sea fácilmente adaptable a cualquier entorno.

Nuestro producto nos servirá para desarrollar software específico para nuestros posibles clientes a un bajo coste gracias a la unificación de esfuerzos en las herramientas comunes de cada empresa como puede ser la contabilidad, facturación,gestión de terceros, gestión de documentos,etc. y a un modelo de tareas y flujos de trabajo lo suficientemente extenso como para adaptarse a las lógica de negocios más comunes de las PYMES[1] españolas (compraventa, fabricación, reparación, alquiler, servicios, etc).

Nuestro cliente objetivo serán aquellas PYMES que por su reducido tamaño y falta de recursos tendrían que optar por soluciones software comerciales y adaptar su forma de trabajo a estas. Nuestra capacidad de desarrollo ágil nos permitirá desarrollar soluciones específicas a las necesidades de nuestro cliente a un bajo coste.

En resumen:

- Ofrecemos a nuestro cliente un software personalizado y adaptado a su modelo de negocio pero sin llegar a los costes del software a medida.
- Será un sistema abierto a modificaciones y mejoras continuas de tal forma que sea el software el que se adapte al cliente y no el cliente al software como ocurre muchas veces al usar software comercial muy especializado como las herramientas SAP.
- El bajo coste de las herramientas y el servicio de “llave en mano” que daremos a nuestros clientes que podrán empezar a funcionar ágilmente desde el primer momento hará que a la larga sea más rentable que productos gratuitos parecidos como OpenBravo que requieren una fase de aprendizaje y adaptación a la empresa hasta que empiezan a ser productivos.

En primera instancia se plantea que la adaptación de las herramientas sea manual por parte del personal cualificado del propio equipo de desarrollo pero no se descarta el diseño de aplicaciones que permitan añadir funcionalidades al cliente final.

Sistema de gestión de empresas enfocado a la nube

Hoy en día estamos viviendo la transición de los sistemas informáticos al cloud-

1 *Acrónimo de pequeña y mediana empresa*

computing o computación en la nube. Poco a poco empiezan a quedar atrás los tiempos en los que cada usuario era propietario de un hardware y un software que debía adquirir, instalar, mantener y actualizar a medida que los requisitos de las nuevas tecnologías crecían y cada usuario era responsable de mantener sus sistemas informáticos libres de virus, proteger sus datos con backups, etc. Todo apunta a que los sistemas informáticos del futuro a nivel de usuario consistirán básicamente en terminales ligeros conectados a internet que nos permitirán subir información a la nube para tenerla accesible en todo momento y desde cualquier otro lugar o dispositivo. Serán terminales sin almacenamiento local ya que todos nuestros datos estarán en la nube y sin especiales capacidades de cómputo (thin client o cliente ligero) ya que esa tarea recaerá en los servidores que formarán esa nube.

En este aspecto, nuestro sistema nace ya preparado para la nube. Nuestro sistema está pensado para depositar toda la carga de trabajo en servidores web, bases de datos y sistemas de fichero en nube y brinda una interfaz web con la que poder añadir y editar toda la información de nuestra empresa y almacenarla en la nube para luego poder acceder a ella, nosotros o cualquier otro usuario autorizado.

2.2 Alcance del software

Base de datos

Se creará una base de datos relacional que permita incluir toda la información necesaria para el negocio. Se crearán todas las entidades necesarias y se relacionarán entre sí de forma coherente y manejable.

Herramientas de gestión

Se tendrán en cuenta todas las necesidades comunes a cualquier empresa y la información manejada en cada caso para el desarrollo de aplicaciones que permitan su tratamiento.

2.3 Capacidades generales

Se crearán herramientas de gestión que faciliten la labor de las tareas más comunes en el entorno empresarial:

Herramientas de contabilidad:

Se desarrollarán herramientas que permitan la gestión del plan general contable para PYMES. Por su extensión se empezará con una gestión básica de los libros contables y operaciones más comunes.

Herramientas de gestión de terceros:

Se desarrollarán herramientas que permitan introducir y gestionar los datos de los terceros con los que nuestro cliente tendrá relación comercial. La herramienta permitirá la gestión de empresas que actúen como clientes o proveedores, entidades bancarias, empleados, etc.

Herramientas de gestión de activos:

El sistema permitirá gestionar sus activos independientemente de su naturaleza. En este aspecto se aportarán soluciones para la gestión de:

- Catálogos de productos: Que nos permita listar, clasificar y documentar los productos manejados en el negocio de nuestro cliente, ya sean materias primas, productos terminados, consumibles, etc.
- Gestión de activos: Herramientas que permitan la gestión de almacenes y stocks.
- Tarificación: Se deberá dar soporte a la creación de tarifas para gestionar los precios de los activos tanto en su compra como en su venta.

Gestión de documentos:

El sistema permitirá almacenar no sólo la documentación típica de cualquier labor comercial (facturas, albaranes, pedidos, etc) sino cualquier otro documento o fichero necesario para el funcionamiento de la empresa.

Gestión de proyectos, tareas y flujos de trabajo:

El mayor esfuerzo para adaptar las herramientas a la lógica de negocio de nuestros futuros clientes consistirá en extrapolar cada una de las actividades llevadas a cabo por el personal de nuestro cliente a un modelo de tareas y flujos de trabajos. El sistema ayudará a gestionar todas estas tareas desde el momento en que se crean hasta el momento en que se cierran e incluirán toda la información necesaria en el proceso.

Por cada tarea se definirá:

- Descripción de la tarea.
- Quién puede solicitar dicha tarea (rol del creador).
- Quién puede realizar la tarea (rol del realizador).
- Qué datos o documentos se deben aportar para poder realizar la tarea (Documento de entrada).

- Qué datos o documentos se generarán y habrá que introducir en el sistema una vez finalizada la tarea (Documento de salida).

Una vez definidas las tareas, sus documentos de entrada y sus documentos de salida se diseñarán los flujos de trabajo. Después de estudiar a fondo la metodología de trabajo del cliente se establecerá quién, cuándo y cómo se crearán las tareas identificadas.

El origen de las tareas puede ser de dos tipos:

- Como consecuencia de un evento externo al sistema como puede ser una solicitud de trabajo.
- Como consecuencia de nuevas necesidades generadas en el transcurso de otra tarea.

De este modo se definirá una serie de tareas “padre” que se podrán generar a partir de sucesos externos y una número ilimitado de tareas “hijo” que se podrán desencadenar a partir de ella con el fin de realizar un trabajo completo y facturable por parte de nuestro cliente.

2.4 Restricciones generales

En este aspecto, nuestro propósito nos obliga a dividir las restricciones. Por un lado debemos pensar en las restricciones obligadas por nuestra intención de desarrollar un software genérico y ágil (Punto de vista de desarrollo) y por otro lado las restricciones originadas por el hecho de no conocer a priori las necesidades de nuestro cliente (punto de vista del cliente)

Punto de vista del del desarrollo:

Necesitaremos una base de datos muy genérica y reutilizable. Deberá contener el máximo de información pero con un mínimo de entidades. Para conseguirlo tendremos que reutilizar y combinar conceptos como cliente y proveedor, empleado, documento/factura/albarán, etc.

En nuestras herramientas deberemos tener siempre bien identificado y separado qué forma parte del núcleo de nuestro del sistema y qué forma parte de las adaptaciones hechas para nuestros clientes. A esta división que haremos en todas las herramientas las llamaremos núcleo y extensiones.

Aunque intentaremos tener un núcleo lo suficientemente genérico para ser utilizado en cualquier entorno se prevé la necesidad de tener que extenderlo, por lo tanto necesitaremos poder actualizar nuestro núcleo sin interferir en las extensiones y viceversa.

Punto de vista del cliente:

A priori no contamos con mucha información sobre nuestro cliente objetivo ni de su entorno, esto nos obliga a tener en cuenta una serie de restricciones:

- Debe ser multiplataforma.
- Debe ser distribuido ya que no conocemos ni el tamaño de la empresa ni su distribución geográfica. Esto implica que sea accesible a través de Internet
- Debe permitir interfaces móviles como tablets o smartphones ya que puede ser un requisito impuesto por nuestro cliente.
- Debe ser escalable y no contener límites en el número de registros, formato de la información manejada, etc.
- Debe ser manejable e intuitivo para ser utilizado por un amplio abanico de perfiles de usuarios finales.

2.5 Características de los usuarios: roles y capacidades

La seguridad será un punto clave del proyecto ya que deberá poder ser accesible a través de Internet y el contenido de los datos almacenados será muy sensible.

Se creará un modelo de usuarios, roles y permisos que permitan gestionar los accesos de los usuarios de forma dinámica. No obstante, como objetivo básico para este proyecto, se plantea que la gestión de roles y usuarios sea parte de la personalización llevada a cabo por el equipo de desarrollo para más adelante delegar y facilitar esta gestión al usuario final añadiendo la figura del administrador.

Pensando en adaptarse a las empresas más pequeñas, el modelo de roles permitirá a un mismo usuario ejercer diferentes roles del mismo modo que para las empresas más grandes varios usuarios podrán pertenecer al mismo rol.

Para el diseño, habrá que tener en cuenta los roles básicos de cualquier entorno empresarial y sus funciones en cada herramienta.

Herramientas de contabilidad (departamento contable)

- Creación, eliminación y modificación de cuentas contables.
- Creación, eliminación y modificación de asientos contables.
- Creación de informes (balance de situación, pérdidas y ganancias, declaración de IVA, etc)
- Apertura y cierre de ejercicios.

- Resolución de incidencias y descuadres.

Herramientas de gestión de terceros: (departamento de administración y recursos humanos)

- Creación, eliminación y modificación de datos de terceros.
- Alta, baja y modificación del personal.
- Introducción, creación y gestión de documentos (facturas entrantes y salientes, albaranes, etc.)

Herramientas de gestión de activos

Departamento de marketing

- Alta, baja y modificación de catálogos.
- Alta, baja y modificación de activos.
- Creación y modificación de tarifas.

Departamento de logística

- Gestión de movimientos y entradas en almacenes.
- Control de stock.

Gestión de proyectos, tareas y flujos de trabajo.

Todas las tareas que identifiquemos con el cliente y que formen parte de su modelo de trabajo se estudiarán para definir quién puede abrir nuevas tareas de ese tipo y a quién se le pueden asignar dichas tareas. En el proceso de estudio del cliente para el que adaptamos nuestras herramientas deberemos completar una tabla como la siguiente (Ejemplo de la empresa de reparación de equipos médicos:

Tarea	**Roles de creación**	**Roles de ejecución**	**Subtareas generadas**
Recepción de equipo para reparar		Logística	Reparación de equipo Valoración de reparación
Reparación de equipo	Logística	Taller	Petición de repuesto Facturación (final de reparación)
Valoración de	Logística	Taller	Elaboración de

reparación			presupuesto
Elaboración de presupuesto	Administración	Administración	Reparación de equipo
Petición de repuesto	Taller	Logística	Reparación de equipo
Facturación (final de reparación)	Taller	Administración	
...	...	...	

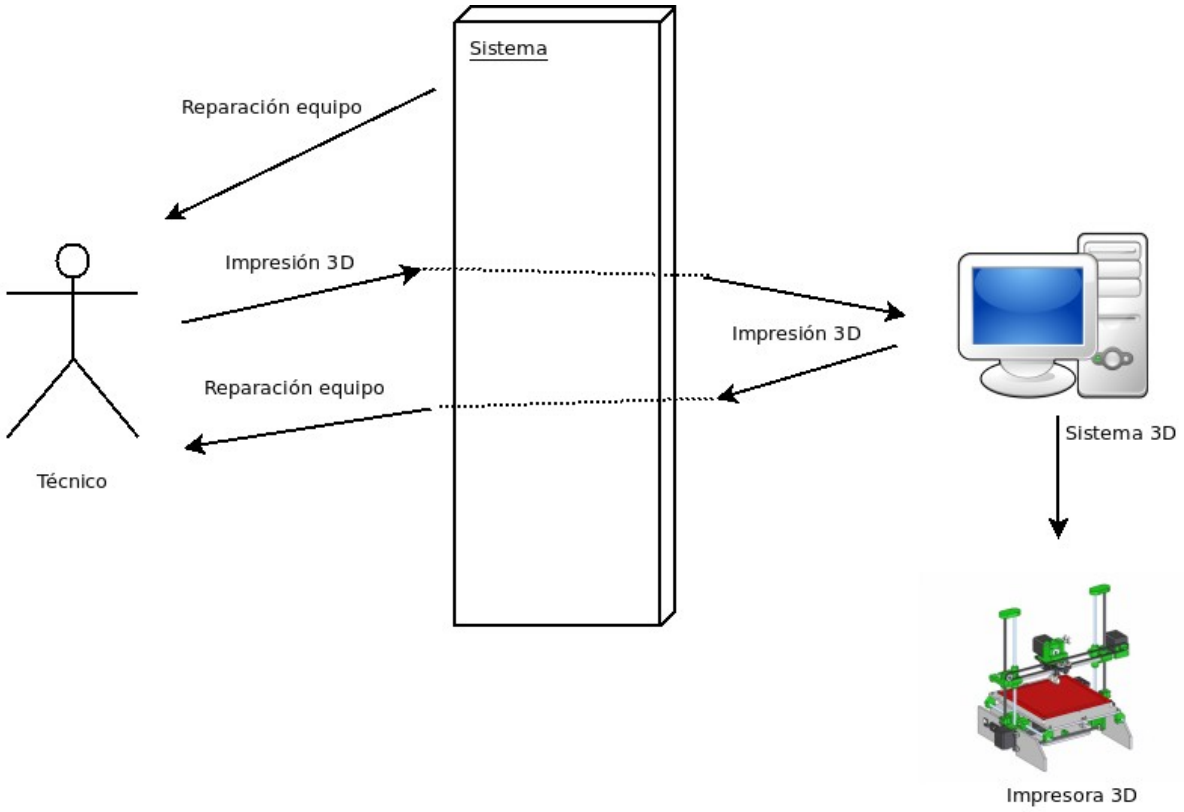
2.6 Entorno operacional

El entorno operacional también será el que marque nuestro cliente. Nuestro paquete básico de herramientas están orientadas a interactuar con personal de nuestro cliente mediante interfaces reproducibles desde diferentes dispositivos móviles o fijos, pero como hemos defendido hasta ahora nuestro objetivo es poder personalizar nuestro software para que se adapte al cliente y no al contrario y es por eso por lo que durante el diseño no se presupone que el rol de ejecución de una tarea tenga que ser una persona sino cualquier otro sistema.

Un ejemplo de lo que nos podría pedir el cliente sería que ante la solicitud de un técnico de una pieza de repuesto este mande imprimir la pieza en una impresora 3D. En nuestra adaptación identificamos lo siguiente:

Tarea	**Roles de creación**	**Roles de ejecución**	**Subtareas generadas**
Reparación de equipo	Logística	Taller	Impresión 3D
Impresión 3D	Taller	Sistema 3D	Reparación de equipo

En este caso el rol de “Sistema 3D” no lo tendría un usuario a través de una interfaz sino un sistema informático capaz de comunicarse con la base de datos para capturar y realizar tareas.


Sistema
Reparación equipo
Impresión 3D
Impresión 3D
Reparación equipo
Técnico
Sistema 3D
Impresora 3D

3 Requisitos

3.1 Justificación de la clasificación de los requisitos

Clasificaremos los requisitos funcionales según la herramienta a la que afectan, teniendo un primer grupo de requisitos de tipo “general” para aquellos requisitos que afectan a todas las herramientas. Por cada tipo de herramienta, agruparemos los requisitos que estén relacionados entre sí por un subtipo.

Los requisitos funcionales irán numerados como RF-<<tipo>>.<<subtipo>>/<<número>>

Los requisitos no funcionales se clasifican por el tipo de componente al que afectan, agrupando igualmente los requisitos relacionados por un subtipo.

Los requisitos no funcionales irán numerados como RNF-<<tipo>>.<<subtipo>>/<<número>>

Las clasificaciones serán:

Requisitos funcionales

RF-1 Generales

RF-1.1 Usuarios

RF-2 Gestión de terceros

RF-2.1 Campos

RF-2.2 Facturación

RF-3 Contabilidad

RF-3.1 Cuentas

RF-3.2 Asientos

RF-3.3 Facturas

RF-3.4 Facturas/Recibos

RF-3.5 Informes

RF-4 Administración

RF-4.1 Empleados

RF-5 Catálogos, productos y tarifas

RF-5.1 Productos

RF-5.2 Catálogos

RF-5.3 Tarifas

RF-6 Activos y almacenes

RF-6.1 Activos

RF-6.2 Almacenes

RF-7 Tareas

Requisitos no funcionales

RNF1 Interfaz

RNF1.1 Usuarios

RNF1.2 Código

RNF1.3 Cliente

RNF1.4 Personalización de herramientas

RNF2 Base de datos

RNF2.1 Modelo de datos

3.2 Requisitos funcionales

3.2.1 Generales

Requisito	RF-1.0/1
Nombre	Interfaz multiusuario.
Tipo	Generales.
Subtipo	
Descripción	Las herramientas deberán ser multiusuario.

Requisito	RF-1.1/1
Nombre	Usuarios concurrentes.
Tipo	Generales.
Subtipo	Usuarios.
Descripción	Varios usuarios podrán interactuar al mismo tiempo con las mismas herramientas.

Requisito	RF-1.1/2
Nombre	Acceso de usuario.
Tipo	Generales.
Subtipo	Usuarios.

Descripción	El acceso a las herramientas serán controlados por usuario y contraseña.

Requisito	RF-1.1/3
Nombre	Roles de usuario.
Tipo	Generales.
Subtipo	Usuarios.
Descripción	Cada usuario tendrá definidos unos roles (descritos previamente por el cliente al equipo de desarrollo) que definirán qué tareas puede realizar y qué información puede ver o modificar.

3.2.2 Gestión de terceros

Requisito	RF-2.0/1
Nombre	Alta/Baja/Edición de terceros.
Tipo	Gestión de terceros.
Subtipo	
Descripción	El sistema permitirá dar de alta y de baja así como modificar los campos de los terceros.

Requisito	RF-2.0/2
Nombre	Número de registros de terceros.
Tipo	Gestión de terceros.
Subtipo	
Descripción	No habrá límite en el número de terceros.

Requisito	RF-2.1/1
Nombre	Campos de terceros.
Tipo	Gestión de terceros.
Subtipo	Campos.

Descripción	• Nombre (Texto).
	• CIF/NIF (texto).
	• Alias (Texto).
	  • Podrá tener varios valores.
	  • Cada alias contendrá un “tipo” (Texto) y un “alias” (Texto).
	• Razón social (Texto).
	• Empleados
	  • Varias direcciones. (Compuesto por dirección, localidad, código postal y provincia) (Texto)
	  • Varios correos electrónicos.
	  • Varios teléfonos.
	• Varias cuentas bancarias
	  • Nombre identificativo. (Texto)
	  • Formato de la cuenta bancaria (ccc / IBAN).
	  • Entidad bancaria. (Texto)
	  • Número. (Texto)
	• Varios departamentos.
	  • Varias direcciones. (Compuesto por dirección, localidad, código postal y provincia)Varios correos electrónicos.
	  • Varias webs.
	  • Varios teléfonos.

Requisito	RF-2.1/2
Nombre	CIF/NIF único.
Tipo	Gestión de terceros.
Subtipo	Campos.
Descripción	No podrá haber dos terceros con el mismo CIF/NIF.

Requisito	RF-2.1/3
Nombre	Alias únicos.
Tipo	Gestión de terceros.
Subtipo	Campos.

Descripción	No se podrá repetir ningún alias de un tercero de tal forma que este será un valor identificativo.

Requisito	RF-2.2/1
Nombre	Datos de facturación.
Tipo	Gestión de terceros.
Subtipo	Facturación.
Descripción	Para poder facturar a un tercero debe haberse introducido previamente los datos fiscales de este (CIF, dirección).

Requisito	RF-2.2/2
Nombre	Cuenta de cliente para facturar.
Tipo	Gestión de terceros.
Subtipo	Facturación.
Descripción	Para poder facturar a un tercero, este deberá tener una cuenta de cliente asociada.

Requisito	RF-2.2/3
Nombre	Cuenta de proveedor para facturas recibidas.
Tipo	Gestión de terceros.
Subtipo	Facturación.
Descripción	Para poder introducir una factura de un tercero, este deberá tener una cuenta de proveedor o de acreedor asociada.

3.2.3 Gestión contable

Requisito	RF-3.1/1
Nombre	PGC 2007.
Tipo	Contabilidad.
Subtipo	Cuentas.

Descripción	El sistema contará con una base de datos de todas las cuentas existentes en el actual plan general de contabilidad para PYMES de 2007.

Requisito	RF-3.1/2
Nombre	Alta/Baja/Edición de cuentas.
Tipo	Contabilidad.
Subtipo	Cuentas.
Descripción	El sistema permitirá dar de alta y de baja así como modificar las cuentas.

Requisito	RF-3.1/3
Nombre	Número de registros de cuentas.
Tipo	Contabilidad.
Subtipo	Cuentas
Descripción	No habrá límite en el número de cuentas.

Requisito	RF-3.1/4
Nombre	Borrado de cuentas.
Tipo	Contabilidad.
Subtipo	Cuentas.
Descripción	No se podrá borrar una cuenta sin antes borrar todos los asientos involucrados.

Requisito	RF-3.1/5
Nombre	Id cuenta automático.
Tipo	Contabilidad.
Subtipo	Cuentas.
Descripción	Al crear una cuenta se le asignará un id único para cada grupo de cuentas del P.G.C.

Requisito	RF-3.1/6
Nombre	Cuentas de terceros.
Tipo	Contabilidad.
Subtipo	Cuentas.
Descripción	A cada cuenta se le podrá asignar un tercero y sólo uno.

Requisito	RF-3.1/7
Nombre	Terceros con cuentas de diferentes grupos.
Tipo	Contabilidad.
Subtipo	Cuentas.
Descripción	Un mismo tercero puede tener varias cuentas asociadas pero no puede tener dos cuentas del mismo grupo.

Requisito	RF-3.2/1
Nombre	Alta/Baja/Edición de asientos.
Tipo	Contabilidad.
Subtipo	Asientos.
Descripción	El sistema permitirá dar de alta y de baja así como modificar los apuntes de cada asiento.

Requisito	RF-3.2/2
Nombre	Número de registros de asientos.
Tipo	Contabilidad.
Subtipo	Asientos.
Descripción	No habrá límite en el número de asientos.

Requisito	RF-3.2/3
Nombre	Principios contables.
Tipo	Contabilidad.

Subtipo	Asientos.
Descripción	No se podrán realizar operaciones contrarias a los principios contables (por ejemplo, principio de no compensación).

Requisito	RF-3.2/4
Nombre	Edición de asientos del ejercicio abierto.
Tipo	Contabilidad.
Subtipo	Asientos.
Descripción	Sólo se podrán crear y editar asientos del ejercicio abierto

Requisito	RF-3.2/5
Nombre	Asientos no cuadrados.
Tipo	Contabilidad.
Subtipo	Asientos.
Descripción	Se permitirán asientos que no sumen cero pero el sistema alertará de este suceso para que sea corregido.

Requisito	RF-3.2/6
Nombre	Cada apunte con cuenta diferente en un mismo asiento.
Tipo	Contabilidad.
Subtipo	Asientos.
Descripción	Un asiento podrá involucrar a tantas cuentas como sea necesario pero en un mismo asiento no podrán aparecer apuntes que hagan referencia a la misma cuenta. En ese caso se sumarán los saldos y se introducirá un único apunte en el asiento.

Requisito	RF-3.3/1
Nombre	Campos de las facturas emitidas.
Tipo	Contabilidad.
Subtipo	Facturas.

Descripción	• Número (Numérico) automático. • Fecha de factura. • Fecha de vencimiento. • Forma de pago (Texto) • Fecha de pago estimado. • Estado (pagado/no pagado)

Requisito	RF-3.3/2
Nombre	Número de registros de facturas.
Tipo	Contabilidad.
Subtipo	Facturas.
Descripción	No habrá límite en el número de facturas.

Requisito	RF-3.3/3
Nombre	Número de factura.
Tipo	Contabilidad.
Subtipo	Facturas.
Descripción	El número de factura será consecutivo y no se podrá repetir en un mismo ejercicio.

Requisito	RF-3.3/4
Nombre	Vencimiento de pago.
Tipo	Contabilidad.
Subtipo	Facturas.
Descripción	El vencimiento de pago será posterior a la fecha de la factura.

Requisito	RF-3.4/1
Nombre	Recibos domiciliados.
Tipo	Contabilidad.
Subtipo	Facturas/Recibos.

Descripción	El sistema facilitará la gestión de remesas de recibos domiciliados en cuenta de las facturas cuyo campo “forma de pago” sea “recibo domiciliado”.

Requisito	RF-3.4/2
Nombre	Cuaderno 19.
Tipo	Contabilidad.
Subtipo	Facturas/Recibos.
Descripción	El sistema creará las remesas de recibos domiciliados de acuerdo con la norma 19 de la AEB.

Requisito	RF-3.5/1
Nombre	Impresión de facturas.
Tipo	Contabilidad.
Subtipo	Informes.
Descripción	El sistema permitirá imprimir o guardar en PDF las facturas para su envío.

Requisito	RF-3.5/2
Nombre	Balance de situación.
Tipo	Contabilidad.
Subtipo	Informes.
Descripción	El sistema ayudará a generar el balance de situación.

Requisito	RF-3.5/3
Nombre	Informe de pérdidas y ganancias.
Tipo	Contabilidad.
Subtipo	Informes.
Descripción	El sistema ayudará a generar el informe de pérdidas y ganancias.

Requisito	RF-3.5/4
Nombre	Cierre de ejercicios.
Tipo	Contabilidad.
Subtipo	Informes.
Descripción	El sistema ayudará a generar los asientos de cierre del ejercicio contable.

Requisito	RF-3.5/5
Nombre	Modelos AT.
Tipo	Contabilidad.
Subtipo	Informes.
Descripción	El sistema ayudará a generar los modelos 111, 115, 303 y 347 de la Agencia tributaria.

3.2.4 Administración

Requisito	RF-4.1/1
Nombre	Alta/Baja/Edición de empleados.
Tipo	Administración.
Subtipo	Empleados.
Descripción	El sistema permitirá dar de alta y de baja así como modificar los campos de los empleados.

Requisito	RF-4.1/2
Nombre	Número de registros de empleados.
Tipo	Administración.
Subtipo	Empleados.
Descripción	No habrá límite en el número de empleados.

Requisito	RF-4.1/3

Nombre	Datos de empleados.
Tipo	Administración.
Subtipo	Empleados.

Descripción	• Nombre (Texto).
	• NIF (texto).
	• Alias (Texto).
	  • Podrá tener varios valores.
	  • Cada alias contendrá un “tipo” (Texto) y un “alias” (Texto).
	• Empleados
	  • Varias direcciones. (Compuesto por dirección, localidad, código postal y provincia)
	  • Varios correos electrónicos.
	  • Varios teléfonos.
	• Varias cuentas bancarias
	  • Nombre identificativo.(Texto)
	  • Formato de la cuenta bancaria (ccc / IBAN).
	  • Entidad bancaria. (Texto)
	  • Número. (Texto)
	• Nóminas
	  • Salario base actual. (el histórico estará disponible en los libros contables)
	  • Cotización C.C. (Porcentaje)
	  • Desempleo. (Porcentaje)
	  • IRPF. (Porcentaje)
	  • Número de pagas.

Requisito	RF-4.1/4
Nombre	Contabilización de nóminas.
Tipo	Administración.
Subtipo	Empleados.
Descripción	El sistema ayudará a crear los asientos contables relacionados con las nóminas de los empleados a partir de sus datos.

Requisito	RF-4.1/5
Nombre	NIF único.
Tipo	Administración.
Subtipo	Empleados
Descripción	El NIF no se podrá repetir entre los empleados.

3.2.5 Gestión de catálogos, productos y tarifas.

Requisito	RF-5.1/1
Nombre	Lista única de productos.
Tipo	Catálogos.
Subtipo	Productos.
Descripción	Solo habrá una lista de productos.

Requisito	RF-5.1/2
Nombre	Campos de los productos.
Tipo	Catálogos.
Subtipo	Productos.
Descripción	• Id (numérico). Se asignará automáticamente. • Tipo. (Texto) • Marca. (Texto) • Modelo. (Texto) • Descripción. (Texto)

Requisito	RF-5.1/3
Nombre	Número de registros de productos.
Tipo	Catálogos.
Subtipo	Productos.
Descripción	No habrá límite en el número de productos.

Requisito	RF-5.1/4
Nombre	Tipo/Marca/Modelo como identificador.
Tipo	Catálogos.
Subtipo	Productos.
Descripción	La terna tipo, marca y modelo será un id único.

Requisito	RF-5.2/1
Nombre	Alta/Baja/Edición de catálogos.
Tipo	Catálogos.
Subtipo	Catálogos.
Descripción	El sistema permitirá dar de alta y de baja catálogos así como modificar la lista de productos que incluye.

Requisito	RF-5.2/2
Nombre	Campos de los catálogos.
Tipo	Catálogos.
Subtipo	Catálogos.
Descripción	• Nombre identificativo único. (Texto) • Lista de productos. • Tercero.

Requisito	RF-5.2/3
Nombre	Número de registros de catálogos.
Tipo	Catálogos.
Subtipo	Catálogos.
Descripción	No habrá límite en el número de catálogos.

Requisito	RF-5.2/4
Nombre	Inserción previa de productos para catálogos.

Tipo	Catálogos.
Subtipo	Catálogos.
Descripción	Para añadir un producto a un catálogo previamente debe haberse introducido el producto en el sistema.

Requisito	RF-5.3/1
Nombre	Alta/Baja/Edición de tarifas.
Tipo	Catálogos.
Subtipo	Tarifas.
Descripción	El sistema permitirá crear y modificar los datos de las tarifas.

Requisito	RF-5.3/2
Nombre	Campos de las tarifas.
Tipo	Catálogos.
Subtipo	Tarifas.

Descripción	• Nombre identificativo único. (Texto)
	• Lista de productos
	  • Importe. (Numérico)
	  • I.V.A. (Numérico)
	  • Descuento. (Porcentaje)
	  • Fecha inicio. Opcional, fecha a partir de la cual es aplicable la tarifa.
	  • Fecha fin. Fecha a partir de la cual no es aplicable la tarifa.
	  • Tercero.

Requisito	RF-5.3/3
Nombre	Número de registros de tarifas.
Tipo	Catálogos.
Subtipo	Tarifas.

Descripción	No habrá límite en el número de tarifas.

Requisito	RF-5.3/4
Nombre	Inserción previa de productos para tarifas.
Tipo	Catálogos.
Subtipo	Tarifas.
Descripción	Para añadir un producto a una tarifa previamente debe haberse introducido el producto en el sistema.

3.2.6 Gestión de activos y almacenes

Requisito	RF-6.1/1
Nombre	Alta/Baja/Edición de activos.
Tipo	Activos y almacenes.
Subtipo	Activos.
Descripción	El sistema permitirá gestionar la lista de activos identificados por nuestro cliente.

Requisito	RF-6.1/2
Nombre	Campos de los activos.
Tipo	Activos y almacenes.
Subtipo	Activos.
Descripción	• Id automático. (Numérico). • Número de serie. (Texto)

Requisito	RF-6.1/3
Nombre	Número de registros de activos.
Tipo	Activos y almacenes.
Subtipo	Activos.
Descripción	No habrá límite en el número de activos.

Requisito	RF-6.1/4
Nombre	Inserción previa de productos para activos.
Tipo	Activos y almacenes.
Subtipo	Activos.
Descripción	Para añadir un activo previamente debe haberse introducido el producto en el sistema.

Requisito	RF-6.2/1
Nombre	Alta/Baja/Edición de almacenes.
Tipo	Activos y almacenes.
Subtipo	Almacenes.
Descripción	El sistema permitirá dar de alta y de baja así como modificar la información de los almacenes.

Requisito	RF-6.2/2
Nombre	Campos de los almacenes.
Tipo	Activos y almacenes.
Subtipo	Almacenes.
Descripción	• Nombre identificativo (Texto). • Ubicaciones: Lista de posibles ubicaciones dentro del almacén. • Tercero. • Dirección.

Requisito	RF-6.2/3
Nombre	Número de registros de almacenes.
Tipo	Activos y almacenes.
Subtipo	Almacenes.
Descripción	No habrá límite en el número de almacenes.

Requisito	RF-6.2/4
Nombre	Activos en almacenes.
Tipo	Activos y almacenes.
Subtipo	Almacenes.
Descripción	El sistema permitirá asociar varios activos a un almacén pero un activo no podrá estar en dos almacenes a la vez o en un mismo almacén dos veces.

3.2.7 Gestión de tareas

Requisito	RF-7.1/1
Nombre	Campos de las tareas
Tipo	Tareas
Subtipo	Tareas
Descripción	• ID único automático. • Tipo (definido por el cliente). • Título (Texto) • Fecha de creación. • Fecha de apertura. • Fecha de cierre . • Fecha de vencimiento (Para indicar la fecha límite para realizar la tarea). • Estado (Nuevo/Abierto/Cerrado/Bloqueado). • Creador. Quien solicita la tarea. • Rol del ejecutor. Definirá el tipo de rol al que debe pertenecer el usuario para ejecutar dicha tarea. • Ejecutor. A quien se le asigna la tarea. • Documento entrada. • Documento salida. • Urgencia (Alta/Media/Baja): Será definido por quien pide la

	tarea. • Prioridad (Alta/Media/Baja). Será una clasificación dada por el usuario al que se le asigna la tarea.

Requisito	RF-7.1/2
Nombre	Número de registros de tareas.
Tipo	Tareas.
Subtipo	Tareas.
Descripción	No habrá límite en el número de tareas.

Requisito	RF-7.1/3
Nombre	Documentos.
Tipo	Tareas.
Subtipo	Tareas.
Descripción	Los documentos podrán ser de cualquier tipo.
Requisito	RF-7.1/4
Nombre	Asignación de tareas.
Tipo	Tareas.
Subtipo	Tareas.
Descripción	Cada usuario tendrá acceso a la lista de tareas pendientes que pueda capturar (es decir tareas marcadas con alguno de los roles al que pertenece dicho usuario) y que no estén ya asignadas. Cuando un usuario decida abrir una tarea se le asignará a él y a partir de este momento desaparecerá de la lista de “no asignadas” y aparecerá en la lista de tareas asignadas a ese usuario (que sólo verá el) hasta que la marque como finalizada. En principio no se contempla la opción de desasignar una tarea ya que, como se explicará más adelante, se podrá simular esta opción diseñando el flujo debidamente.

Requisito	RF-7.1/5

Nombre	Administración de tareas.
Tipo	Tareas.
Subtipo	Tareas.
Descripción	En todo momento, un usuario con permisos avanzados podrá realizar correcciones en cuanto al estado de las tareas y a quien están asignadas.

Requisito	RF-7.1/6
Nombre	Información para la tarea.
Tipo	Tareas.
Subtipo	Tareas.
Descripción	Al abrir la tarea el sistema mostrará al usuario la información necesaria para realizar dicha tarea (documento de entrada)

Requisito	RF-7.1/7
Nombre	Cierre de las tareas.
Tipo	Tareas.
Subtipo	Tareas.
Descripción	Al completar la información requerida por cada tarea (documento de salida) el sistema ofrecerá dar por cerrada la tarea.

Requisito	RF-7.1/8
Nombre	Flujos de las tareas.
Tipo	Tareas.
Subtipo	Tareas.
Descripción	Al cerrar una tarea el sistema ofrecerá abrir nuevas tareas según el flujo establecido en la toma de requisitos con el cliente.

Requisito	RF-7.1/9
Nombre	Flujos de las tareas.

Tipo	Tareas.
Subtipo	Tareas.
Descripción	Una tarea podrá desencadenar la apertura de más de una tarea hija.

Requisito	RF-7.1/10
Nombre	Visibilidad de las tareas.
Tipo	Tareas.
Subtipo	Tareas.
Descripción	Un usuario solo verá las tareas asociadas a su rol.

3.3 Requisitos no funcionales

3.3.1 Interfaz

Requisito	RNF-1.0/1
Nombre	Interfaz web común.
Tipo	Interfaz.
Subtipo	
Descripción	Se generará una interfaz web común para todas las herramientas. El lenguaje utilizado será HTML, PHP y JavaScript.

Requisito	RNF-1.1/1
Nombre	Adaptación web al usuario.
Tipo	Interfaz.
Subtipo	Usuarios.
Descripción	Dependiendo del rol del usuario cambiará los menús y las opciones de la interfaz.

Requisito	RNF-1.2/1

Nombre	Visualización de tareas.
Tipo	Interfaz.
Subtipo	Código.
Descripción	En la fase de adaptación de las herramientas a nuestro cliente, el equipo de desarrollo deberá identificar (con ayuda del cliente) los documentos de entrada y salida necesarios para cada tarea. El formato de estos documentos serán definidos por las necesidades de nuestro cliente y por cada tipo de documento el equipo de desarrollo generará un código PHP capaz de mostrar la información del documento en la interfaz web. La generación de este código deberá seguir una serie de normas para que se adapte al resto de herramientas. La idea es que cada vez que en el núcleo de nuestro sistema se necesite mostrar un documento el sistema identifique por el tipo de documento que código debe ejecutar para mostrar la información deseada de forma correcta.

Requisito	RNF-1.2/2
Nombre	Actualización de tareas.
Tipo	Interfaz.
Subtipo	Código.
Descripción	Al igual que hemos explicado para la visualización de las tareas, por cada tipo de documento se generará un formulario en formato HTML que permita introducir la información necesaria. Cada vez que necesitemos editar una tarea el sistema identificará por el tipo de documento qué formulario debe mostrar.

Requisito	RNF-1.2/3
Nombre	Ejecución al cerrar tareas.
Tipo	Interfaz.
Subtipo	Código.
Descripción	Cada tarea podrá tener asociado un código PHP que se ejecutará

	en el momento de cerrar la tarea.

Requisito	RNF-1.3/1
Nombre	Navegador del cliente.
Tipo	Interfaz.
Subtipo	Cliente.
Descripción	El cliente deberá tener un navegador web con posibilidad de imprimir en PDF.

Requisito	RNF-1.4/1
Nombre	Gestión de tareas.
Tipo	Interfaz.
Subtipo	Personalización de las herramientas.
Descripción	Para cada cliente se identificarán los tipos de tareas necesarios y el equipo de desarrollo introducirá la información en la base de datos manualmente.

Requisito	RNF-1.4/2
Nombre	Gestión de flujos.
Tipo	Interfaz.
Subtipo	Personalización de las herramientas.
Descripción	Una vez identificadas las tareas, el cliente definirá los flujos de trabajo y el equipo de desarrollo introducirá la información en la base de datos manualmente.

3.3.2 Base de datos

Requisito	RNF-2.0/1
Nombre	Base de datos relacional.
Tipo	Base de datos.

Subtipo	
Descripción	Toda la información se guardará en una base de datos relacional MySQL.

Requisito	RNF-2.1/1
Nombre	Campo flujo en tarea.
Tipo	Base de datos.
Subtipo	Modelos de datos.
Descripción	Para cada tarea se incluirá un atributo “flujo” para identificar todas las tareas relacionadas en un mismo flujo. Este atributo tendrá como valor el id de la tarea inicial.

Requisito	RNF-2.1/2
Nombre	Entidad “contacto”.
Tipo	Base de datos.
Subtipo	Modelos de datos.
Descripción	Dado que para los datos de los contactos, empleados y los departamentos se almacena la misma información se creará la entidad “contacto” para unificarlo en la base de datos. De esta forma cada empleado y cada departamento tendrá asociado el id del contacto que contiene su información.

Requisito	RNF-2.1/3
Nombre	Tabla de clasificación de cuentas
Tipo	Base de datos.
Subtipo	Modelos de datos.
Descripción	Necesitaremos una tabla para almacenar la clasificación de las cuentas contables de cara a realizar el informe de pérdidas y ganancias y el balance de situación.

Requisito	RNF-2.1/4

Nombre	Campos de documento.
Tipo	Base de datos.
Subtipo	Modelos de datos.
Descripción	• Id • Tipo (texto). • Fecha • Número • Referencia (Texto). • Tercero • Formato (Texto). • Ruta al fichero (Texto). • Metadatos (Texto).

Requisito	RNF-2.1/5
Nombre	Número de documento
Tipo	Base de datos.
Subtipo	Modelos de datos.
Descripción	No podrá haber dos documentos del mismo tipo con el mismo número.

Requisito	RNF-2.1/6
Nombre	Modelos de datos.
Tipo	Tipo de datos de urgencia y prioridad.
Subtipo	Modelos de datos.
Descripción	La urgencia y la prioridad de las tareas se guardará con un valor de 0-100 para facilitar ordenar las tareas. A la hora de mostrar este campo en la interfaz web se sustituirá el valor numérico por los valores citados anteriormente de alta (75), media (50), baja (25) de esta forma será más sencillo mostrar una lista ordenada y tendremos la posibilidad de intercalar nuevos valores (hasta 97 más) si el cliente lo solicita, por ejemplo: muy alta(80), muy

	baja((20).

4 Arquitectura

4.1 Modelo de información

En nuestra especificación de requisitos hemos planteado multitud de tareas comunes a muchos tipos de empresas, pero no debemos olvidar que nuestro objetivo no es únicamente dar una solución puntual a cada uno de estos requisitos sino construir un software y diseñar una estructura de datos lo suficientemente genérica para que por sí sola nos permita adaptar nuestro software a nuevos requisitos planteados por nuevos clientes o incluso modificar los requisitos ya planteados con nuevas restricciones.

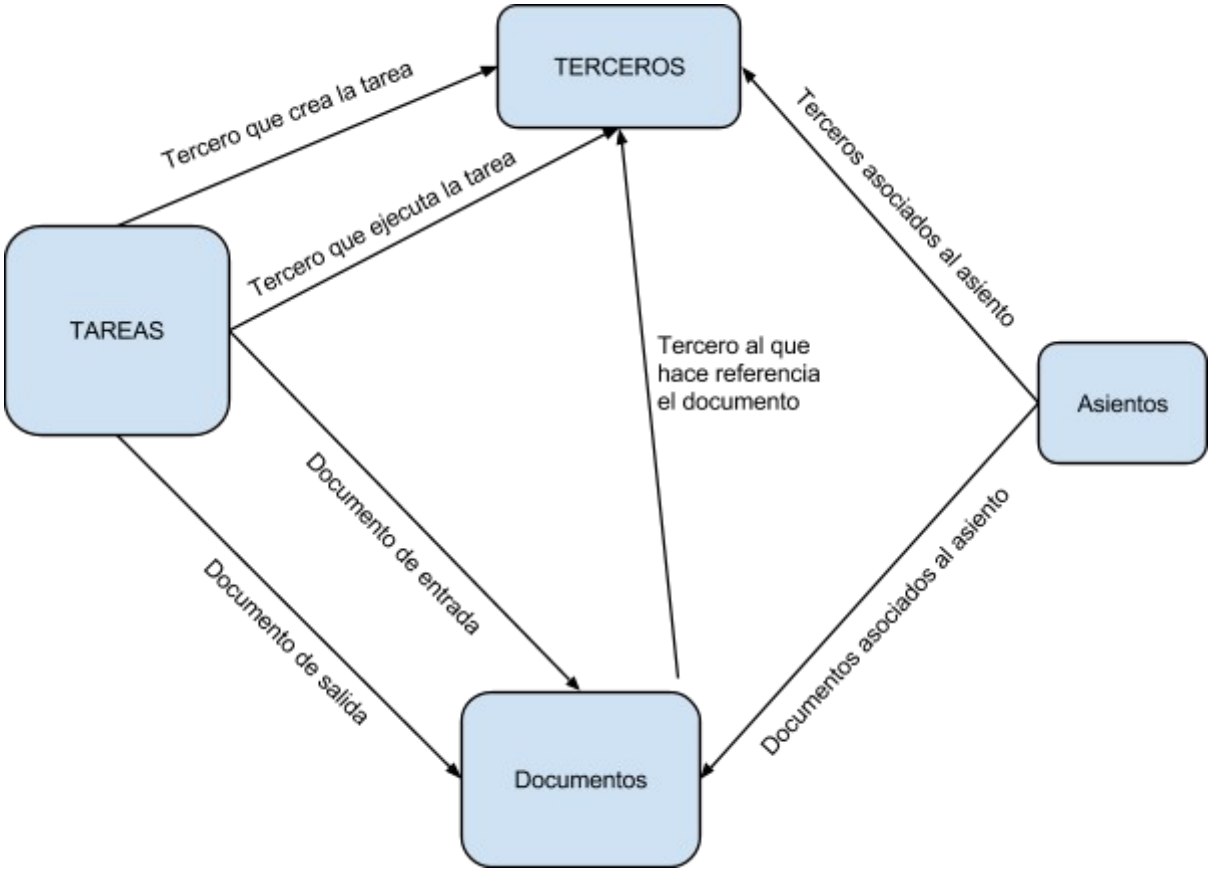
No queremos empezar a desarrollar infinidad de componentes para ir cumpliendo cada uno de los requisitos planteados, lo que buscamos es diseñar los componentes básicos que sustituyan a todos los demás. Después de analizar todos los requisitos y todos los objetos o entidades que se extraen de ellos se ha encontrado una solución que nos permite gestionar todos ellos con un número mínimo de componentes lógicos. Estos son:

- **Documentos:** Es el componente clave que nos permitirá almacenar cualquier tipo de información sea cual sea su naturaleza. Como veremos más adelante será el componente más dinámico de nuestro sistema y el que contendrá gran parte de la lógica de nuestras futuras empresas cliente. **Toda la información que deba ser tratada por nuestro sistema lo convertiremos en un documento.**

- **Tareas:** Será otro componente clave para nuestro sistema ya que lo usaremos para gestionar cualquier acción, tarea o trabajo necesario para el funcionamiento de nuestra empresa cliente. **Cualquier caso de uso identificado lo convertiremos en una tarea.**

- **Terceros:** Más que un componente, es una entidad que nos ayudará a relacionar el resto de nuestros componentes. **Cualquier empresa, persona o entidad que interactúe con nuestra empresa cliente será un tercero.**

- **Libro de cuentas:** Será uno de los pocos componentes que tendrá estructura por sí sólo y no será un documento por motivos que explicaremos con detalle más adelante, pero sí hará uso de documentos para complementar su funcionalidad.



4.1.1 Documentos

Es el componente básico de nuestro sistema y el que contendrá la mayor parte de la información y la lógica de los negocios de nuestras empresas cliente.

En el momento de analizar los requisitos impuestos por nuestros clientes deberemos identificar los bloques de información que necesitaremos almacenar, tratar y dotar de lógica para el funcionamiento del negocio y por cada uno de ellos crearemos la estructura de un documento:

En el momento en el que un nuevo cliente nos pide diseñar un software de gestión para su empresa y nos hace llegar una nueva serie de requisitos deberemos identificar los bloques de información que necesitaremos almacenar, discernir cuáles de ellos ya tenemos resueltos en nuestro sistema base y diseñar la estructura y la lógica para cada nuevo elemento identificado. Por cada elemento o bloque de información (conjunto de datos íntimamente relacionados) crearemos la estructura de un documento:

Un documento se compone principalmente de:

4.1.1.a Datos básicos del documento:

Estos son atributos comunes inherentes a la idea tradicional de un documento y que

nos permitirán clasificar, ordenar y organizar la información que contienen:

Estos atributos, en nuestra primera versión del sistema, serán:

- **Identificador único:** Un ID numérico, autoincremental, que identificará inequívocamente nuestro documento y por lo tanto podremos usar dicho atributo para hacer referencia a él.

- **Tipo:** Identificador alfanumérico que nos dirá el tipo de documento que estamos tratando y con el que podremos saber qué información contiene, cómo está organizada dicha información y que tratamiento le debemos dar en nuestro sistema. Ejemplos de típicos serían “factura”, “albarán”, “recibo”, “nómina”, etc

- **Número de documento:** Al igual que el identificador único, será un número, pero en este caso el valor únicamente se incrementará por cada documento del mismo tipo que creemos. Este número nos será útil para aquellos documentos que deben tener numeración propia. El ejemplo más claro de uso de este identificador será el número de factura, ya por cada factura que emita nuestro cliente se deberá crear un número de factura consecutivo, sin saltos y sin repeticiones. En este aspecto, con el número de documento y el tipo de documento anteriormente explicado obtendremos otro identificador único del documento.

- **Fecha del documento:** La fecha y la hora en la que se crea el documento.

- **Título:** Título del documento y por lo general el nombre con el que se mostrará al usuario en la interfaz gráfica ya que contendrá una breve descripción de la información que contiene.

- **Referencia:** Otro tipo de identificador pero en este caso alfanumérico y sin reglas preestablecidas. Será multiuso y para cada tipo de documento la referencia podrá tener significados diferentes.

- **Tercero:** Cada documento podrá estar asociado a cualquiera de los terceros que tengamos identificados en el sistema. A esta asociación podremos darle un significado diferente dependiendo del tipo de documento que queramos darle. En algunos casos el significado de este valor será casi obligado como por ejemplo a la hora de asociar una factura emitida con el cliente al que va dirigida, en otros casos lo podremos usar para añadir información al documento como por ejemplo el usuario que lo creó y en otros puede que no tenga sentido asignarle ningún valor y dejar el atributo vacío.

- **Estado:** Cadena de caracteres que permite especificar el estado del documento. El contenido de este campo es libre para el desarrollador que diseñe dicho documento. Puede usarse por ejemplo:

- Para marcar documentos como “Elaborándose” o “Finalizado”.
- Marcar facturas como “Pagadas” o “Pendientes de pago”.
- Marcar usuarios como “Bloqueados” o “Activos”.

Es importante recalcar que estos atributos los hemos seleccionado por identificarlos como muy comunes y que previsiblemente serán necesarios en gran parte de los documentos futuros que tengamos que desarrollar para nuestros clientes. De esta forma, en algunos tipos de documentos parte de estos atributos pueden ser imprescindibles (por ejemplo la fecha y el tercero si se trata de un documento de tipo “factura”) y en cambio otros atributos no sean necesarios (una factura no suele llevar título o referencia). Al ser tan comunes merece la pena que tengan entidad propia ya que agilizarán y nos facilitaran el trabajo de diseño futuro.

Del mismo modo, es seguro que estos atributos serán insuficientes para almacenar toda la información requerida por el cliente, no sólo porque necesite asociar más de una fecha, una referencia o un tercero al documento sino porque necesitemos añadir otro tipo de datos (listados, teléfonos, direcciones, etc). Pero como no podemos preverlos todos y añadirlos como atributos básicos, para almacenar toda esta información extra tendremos disponible los metadatos del documento.

4.1.1.b Metadatos

Será un campo de texto en nuestra base de datos relacional que podrá contener cualquier tipo de información y que usaremos para guardar la propia información contenida en el documento. Aunque su contenido es totalmente libre la mayoría de las veces contendrá una lista de atributos clave-valor y por lo tanto se ha decidido que la mejor forma de estructurar dicha información en estos casos es un formato ligero de tipo JSON[2] (JavaScript Object Notation) , fácil y rápido de crear y de leer. JSON nos brinda un sistema abierto para alojar de forma estructurada y ordenada cualquier tipo de información en formato de una cadena de caracteres almacenable en nuestra base de datos relacional.

Estos metadatos podrán contener cualquier tipo de dato como por ejemplo:

- Otras fechas. Ya que, aunque entre los atributos básicos contamos con un campo “fecha de creación” podremos necesitar asociar más fechas al documento (última modificación, revisiones, etc).
- Valores numéricos.
- Cadenas de texto.

2 Ver apartado “Tecnologías usadas”

- Identificadores de cualquier otro componente de nuestro sistema (otros terceros relacionados con el documento, cuentas contables, tareas o incluso referencias a su vez a otros documentos). Las referencias a otros documentos serán de gran utilidad cuando queramos unir documentos y poder tratarlos como uno sólo, por ejemplo, como explicaremos más adelante, cuando necesitemos asignar varios documentos a una tarea.
- Direcciones a sistemas de ficheros de nuestro servidor y que podremos usar para adjuntar cualquier tipo de fichero a nuestro documento.
- URLs que apunten a páginas web, servidores de Webdav, ficheros en google docs, dropbox, etc.
- Listado de cualquiera de los ejemplos citados.
- En definitiva, cualquier dato que queramos almacenar.

4.1.1.c Procesos automatizados

Como hemos explicado anteriormente, los documentos contendrán en sí mismos la mayor parte de la lógica de funcionamiento de nuestras empresas cliente. Una vez identificados los datos contenidos en nuestro documento el segundo paso será dotar al documento de lógica de procesos. Esto lo haremos identificando cada uno de los procesos que deberán ejecutarse al interactuar con nuestros documentos.

A priori, se han identificado las siguientes formas de interactuar con los documentos:

- **Creación del documento:** A la hora de diseñar nuestros documentos podremos añadirle la lógica asociada con la creación del mismo que por lo general estará enfocado a los valores por defecto del mismo. Por ejemplo:
    - Al crear un documento del tipo factura podremos necesitar que los datos del cliente se rellenen automáticamente.
    - Al crear un documento de tipo “inventario de almacén”, queremos que este se rellene automáticamente a partir de la información facilitada por otros tipos de documentos ya existentes del tipo “ albaranes de entrada” o “albaranes de salida”.
- **Guardar el documento:** Principalmente contendrá la lógica para transformar los datos introducidos en la interfaz de usuario en el formato seleccionado para almacenar dicha información en nuestra base de datos, pero podrá contener más procesos que deben ejecutarse a la hora de guardar un documento. Por ejemplo:
    - Al guardar una factura automáticamente se creará el asiento contable

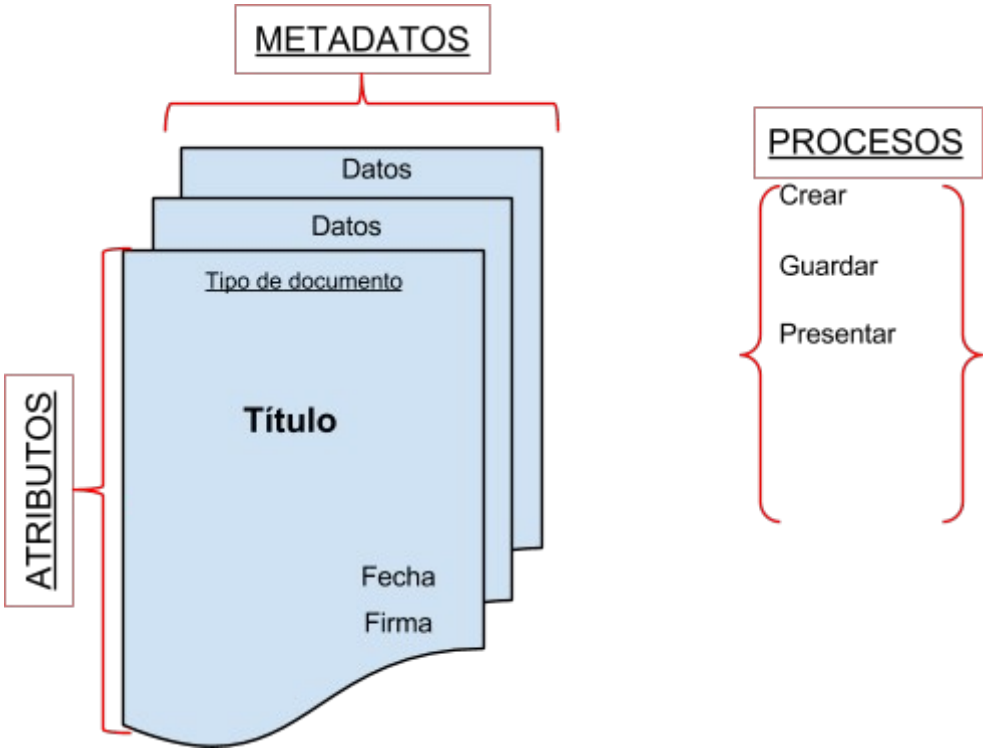
correspondiente.

- Al guardar información fiscal puede que necesitemos modificar cada uno de los documentos de nómina que tengamos en nuestro sistema.
- Al guardar un documento de salida de stock del almacén puede que nos sea útil generar automáticamente una nueva tarea en nuestro sistema para pedir nuevo material.

- **Presentación del documento:** A la hora de visualizar el documento puede que necesitemos mostrarlo de forma diferente dependiendo del motivo para el que lo estamos visualizando. A priori hemos identificado 4 modos de visualización diferentes pero siempre se podrán añadir nuevos:
  - **Visualización en la interfaz:** Organizará la información de forma que sea cómodamente visible en nuestra interfaz de usuario y en la mayoría de los casos permitirá enlazar la información contenida en el propio documento con otros datos creando por ejemplo enlaces a los terceros, cuentas contables, tareas, otros documentos, etc.
  - **Edición del documento:** En este caso, cada uno de los datos que el usuario deba o pueda modificar del documento se deberá convertir en un botón de nuestra interfaz que le permita hacerlo, el conjunto de todos los botones que permiten modificar los metadatos del documento será el formulario del documento. Dichos botones estarán vacíos la mayoría de las veces cuando el documento esté recién creado pero en el caso de abrir para su edición un documento previamente guardado los botones deberán contener el valor actual del atributo correspondiente.
  - **Impresión de documento:** Será un modo de visualización muy común. Por lo general contendrá la misma información que el modo de visualización en la interfaz pero posiblemente enriquecido con adornos o información adicional como cabeceras, logos, datos de la empresa, numeración de páginas, etc.
  - **Descarga de documentos:** Dependiendo del tipo de documento podremos querer que sea descargable desde la interfaz de usuario al almacenamiento local del dispositivo desde el que se accede. El fichero que generará cada documento podrá ser diferente dependiendo del tipo de documento que sea. Así por ejemplo:
    - Para facturas, albaranes y demás documentación añadiremos a nuestro documento la capacidad de convertir dicha información en PDFs o similares (dependerá de nuestra empresa cliente).

- Documentos con información bancaria puede ser útil dotarlos con la lógica necesaria para transformar los datos en estándares fijados por las entidades competentes como los de la AEB (Normas 19,43, etc).

- **Borrar documento**: Contendrá las operaciones necesarias para borrar un documento, a parte de la operación de borrado puede que sea necesario realizar otras acciones, como por ejemplo:
    - Añadir, borrar o modificar asientos contables.
    - Añadir, borrar o modificar otros documentos.
    - Añadir, borrar o modificar tareas.



4.1.2 Tareas

Será otro componente básico, muy utilizado y muy importante en nuestro sistema ya que por cada caso de uso que identifiquemos en nuestra empresa cliente deberemos crear una tarea para tratarlo y es por eso mismo por lo que cuenta con estructura propia.

4.1.2.a Diseño de los flujos de trabajo y tareas

Llamaremos flujo de trabajo a la agrupación de tareas (casos de uso) que identifiquemos como necesarias en nuestro sistema y que guardan relación entre sí.

Como se explicará más adelante un flujo no será una entidad propia porque se tratará como un documento. Para diseñar cada flujo identificaremos tres elementos:

- Las tareas que forman parte del flujo.
- Los saltos que determinarán el modo de avanzar de una tarea a otra dentro del flujo.
- Las acciones, que no serán más que elementos intermedios entre tareas. Estas acciones nos permitirán crear saltos a múltiples tareas en el mismo salto.

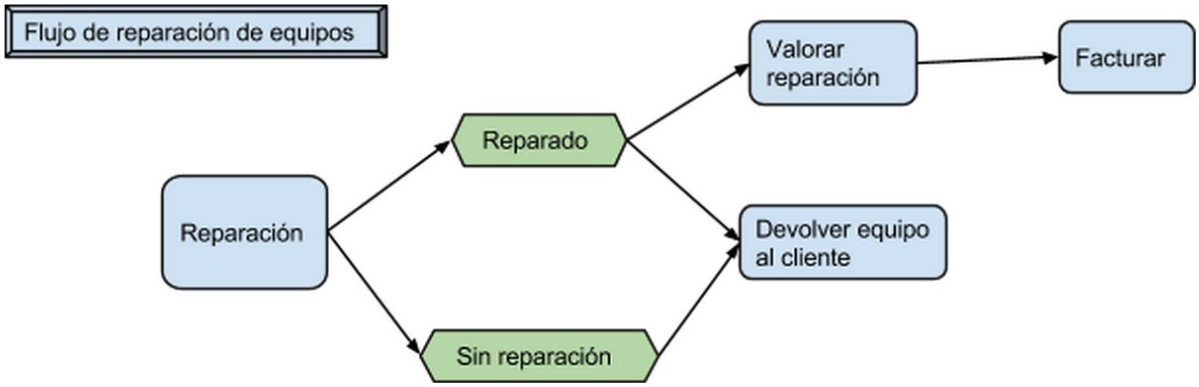
En la siguiente figura se ve un ejemplo de diseño de un flujo de trabajo. En este caso describe las diferentes tareas realizadas por una empresa de reparación de equipos de cualquier tipo, desde que llega el equipo al taller hasta que se le devuelve y factura al cliente.

El cliente nos expone 4 casos de uso:

- **Reparación:** Cuando el técnico se dispone a efectuar la reparación.
- **Valorar reparación:** Una vez reparado el equipo un supervisor de taller deberá analizar el tipo de reparación realizada para escribir una breve explicación de la avería y valorar el gasto que ha supuesto.
- **Facturar:** Con la valoración hecha previamente por parte del supervisor, un empleado de departamento de administración deberá confeccionar la factura.
- **Devolver el equipo:** Una vez que el técnico haya terminado la labor de reparación (con o sin éxito) otro empleado deberá preparar el equipo para ser devuelto a su dueño (empaquetar e imprimir un albarán).

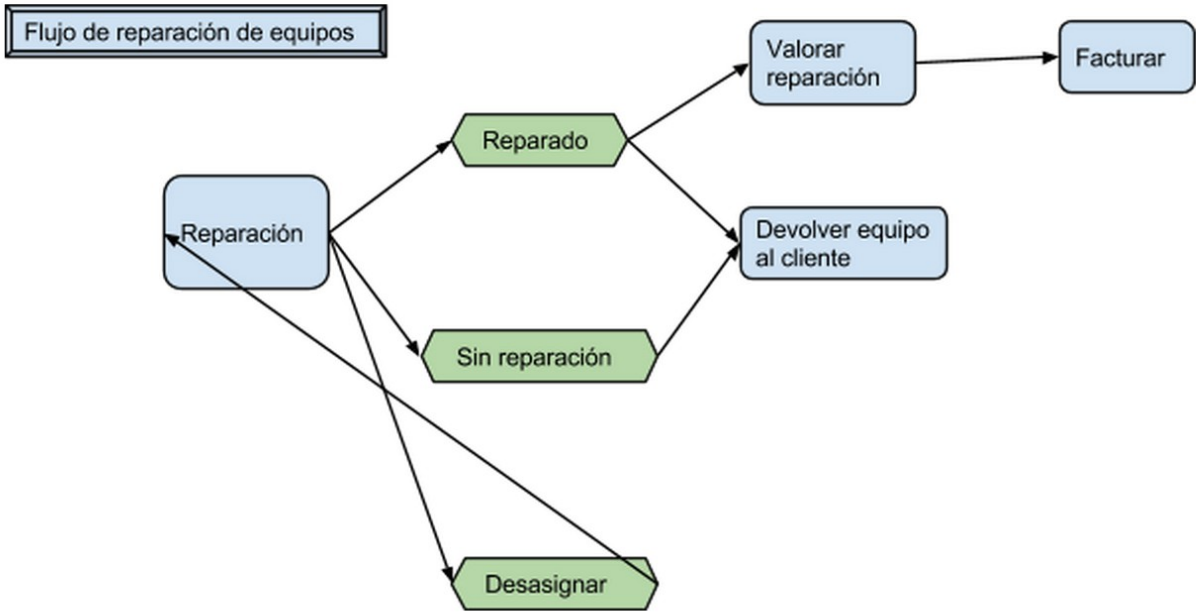
El cliente nos pide una serie de restricciones:

- Sólo se valora y factura una reparación si el equipo se ha reparado con éxito.
- Para confeccionar la factura es imprescindible que el supervisor la haya valorado.
- En cualquier caso, se haya podido reparar o no, el equipo siempre es devuelto al cliente.
- También nos explica que una vez que el equipo ha sido reparado, este debe prepararse para ser devuelto inmediatamente aunque el departamento de administración o el supervisor no hayan hecho aún sus respectivas tareas.



4.1.2.b Acción "desasignar"

Cuando el administrador decida que un usuario que se ha asignado una tarea debe tener la posibilidad de desasignársela será necesario diseñar el flujo correctamente. Para ello, se creará una acción con título "desasignar" cuyo origen sea el tipo de tarea que queremos desasignar y cuyo destino sea esa misma tarea. De esta forma cuando el usuario de por finalizada la tarea con dicha opción se creará una nueva tarea sin asignar del mismo tipo, a efectos prácticos será como si se volviera atrás.



4.1.2.c Atributos de las tareas

Los datos de cada tarea se almacenarán en nuestra base de datos relacional siendo los siguientes:

- **Identificador único:** Será numérico y autoincremental de tal forma que lo podremos usar para relacionarlos con otras tareas, asientos contables, documentos, etc
- **Padre:** En el caso en que la tarea se genere a partir de otra, se guardará el dicha relación con el ID de la tarea padre.
- **Flujo:** Será el identificador único y común a todas las tareas con parentesco. Este identificador nos permitirá identificar rápidamente todas las tareas relacionadas entre si por ser padre, hija, nietas etc, es decir, que pertenezcan al mismo flujo.
- **Creador:** Será una referencia al id del tercero que solicita la tarea.
- **Tipo de flujo-Tarea de flujo:** Esta dupla nos servirá para identificar el tipo de flujo y el paso concreto al que pertenece la tarea en dicho flujo. En nuestro ejemplo anterior esta dupla podría tomar los valores reparacion-solicitud, reparación-valorar, reparación-devolver, reparación-facturar.
- **Título:** Un título descriptivo que le permita al usuario identificar la tarea.
- **Fecha de creación:** Fecha en la que se dio de alta la tarea.
- **Fecha de inicio:** Fecha en la que la tarea fue asignada a un usuario.
- **Fecha fin:** Fecha en la que se cerró la tarea.
- **Vencimiento:** Fecha que podrá usarse para fijar una fecha límite para la realización de la tarea o una fecha concreta en la que debe realizarse dicha tarea.
- **Estado:** Estado actual de la tarea. Podrá tener tantos valores como los requisitos de nuestra empresa cliente determinen. Los más básicos serán, “nuevo” (para tareas recién creadas), “abierto” (para tareas ya asignadas a un tercero), “cerrado” (para tareas finalizadas).
- **Asignado:** Identificador del tercero que captura la tarea o al que se le asigna la tarea y por lo tanto responsable de la realización de la misma.
- **Documento de entrada:** Documento que permitirá aportar la información necesaria para que el usuario correspondiente pueda llevar a cabo la tarea. Como explicamos anteriormente, es posible que necesitemos pasar dos o más documentos de entrada para que el usuario al que se le asigna la tarea tenga a su

disposición toda la información necesaria para realizarla, en ese caso la solución será crear un nuevo tipo de documento, por ejemplo:

Para la tarea de crear un albarán de envío de un equipo a su correspondiente dueño necesitamos por un lado los datos del equipo y por otro lado los datos del dueño. Para los datos de un equipo hemos diseñado un tipo de documento llamado “Equipo”, para los datos de envío hemos creado otro tipo de documento llamado “Dirección de envío”. Para poder asignar ambos documentos a la tarea necesitamos diseñar un nuevo tipo de documento llamado “Datos albarán” cuyos metadatos contendrán la referencia al ID de los otros dos documentos en formato JSON ({“equipo”:”1234”,”dirección”:”5678”}). A la hora de editar un documento de este tipo se mostrará un formulario donde el usuario seleccionará un equipo de una lista de equipos pendientes de enviar y una dirección de una lista de posibles direcciones. A la hora de visualizar este tipo de documento, el sistema recuperará de la base de datos los dos documentos a los que hace referencia y mostrará en pantalla los datos que necesarios para crear un albarán.

- **Documento de salida:** Documento a cumplimentar por el usuario que realizó la tarea y que solicita la información concerniente a la misma y que necesitaremos almacenar en nuestro sistema o que será necesaria para la realización de futuras tareas hija.

El tipo de “documento de entrada” y el tipo de “documento de salida” serán los que realmente identifiquen cada uno de los casos de uso que estamos resolviendo en cada caso. Es decir, siguiendo el ejemplo anterior, con ayuda de nuestro cliente identificamos los siguientes tipos de documentos:

Albarán de recepción: Contendrá los datos del equipo (marca, modelo, serie,...), el origen, propietario y la descripción de la avería o síntomas apreciados por este.

Informe de reparación: Explicación por parte del técnico de las labores realizadas para localizar y reparar la avería.

Valoración de la avería: Incluye el coste real de la reparación y una explicación “menos técnica” que se le dará al dueño del equipo sobre la reparación efectuada.

Factura: Será una factura que contendrá los datos estándar.

Albarán de entrega: Un albarán estándar con los datos del equipo para certificar que el equipo fue devuelto.

Cada uno de estos documentos pueden ser necesarios para realizar algunas de las tareas identificadas anteriormente, es decir, serán el documento de entrada. Del mismo modo, para dar por completada las tareas habrá que rellenar el consiguiente documento

de salida. Por cada una de las 4 tareas ya citadas identificamos cuales deben ser sus documentos de entrada y de salida:

Tarea	Documento de entrada	Documento de salida
Reparación	Albarán de recepción	Informe de reparación
Valorar reparación	Informe de reparación	Valoración de avería
Facturar	Valoración de avería	Factura
Devolver equipo	Albarán de recepción	Albarán de entrega

De esta forma por cada tipo de tarea tendremos un patrón definido por los tipos de documentos de entrada y de salida, es decir podremos crear tantas tareas de reparación como sean necesarias simplemente indicando que el documento de entrada de esa tarea es de tipo “Albarán de recepción” y el documento de salida es de tipo “Informe de reparación”.

4.1.2.d Ciclo de vida de la tarea.

Como se ha descrito hasta ahora, los diferentes casos de uso que la labor comercial de nuestra empresa cliente requiera serán tratados por nuestro sistema como tareas dentro de un flujo de trabajo. Pero dentro del ciclo de vida de cada una de las tareas que forman parte de un flujo de trabajo identificamos varios posibles estados.

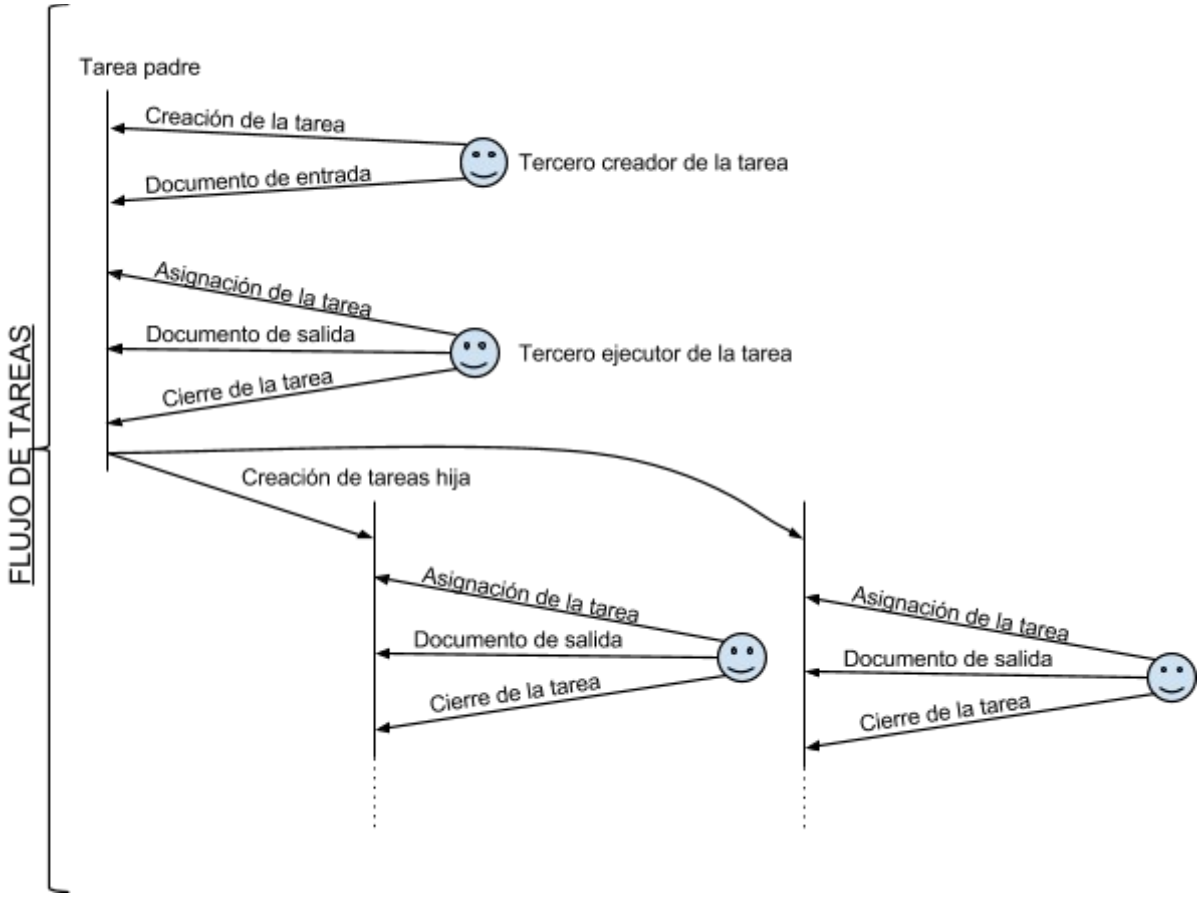
- **Tarea nueva:** Será cuando la tarea esté recién creada, ya sea como tarea de inicio de un nuevo flujo de trabajo o como subtarea generada a partir de otra tarea. En este momento la tarea no tiene tercero asignado que deba realizarla y por tanto se se entiende que aún no se ha realizado ningún trabajo de la misma Las tareas en este estado podrán ser visualizadas por cualquier usuario (con permisos) pudiendo ver tanto los datos de la propia tarea como su documento de entrada, pero no podrán crear ni editar su documento de salida así como no podrán realizar ninguna acción sobre ella.

- **Tarea abierta:** Una vez que la tarea es asignada a un tercero,ésta se considera abierta, se le asigna fecha de inicio y se crea su documento de salida (con los datos por defecto, auto rellenados o en blanco) para que el usuario asignado pueda cumplimentarlo. En este estado, únicamente el usuario al que se le ha asignado la tarea (y los administradores con privilegios suficientes) podrán visualizar dicha tarea. El usuario al que se le asigna la tarea podrá abrir y editar tantas veces como necesite el documento de salida hasta que dé por finalizada y cerrada la tarea.

- **Tarea cerrada:** Una vez finalizada la tarea y siempre y cuando el sistema

valide que el documento de salida está debidamente cumplimentado por parte del usuario, éste podrá dar por cerrada la tarea y en caso de que el diseño del flujo al que pertenece así lo indique, el sistema solicitará al usuario seleccionar el siguiente paso que debe seguirse en el flujo y por lo tanto las tareas nuevas que se deberán crear.

No siempre será necesario pasar por todos los estados de estas tareas, pudiendo saltarse o agruparse varios en uno solo. Por ejemplo, podría ser necesario que una tarea se asigne a un usuario en el mismo momento de su creación o un usuario podría llegar a capturar la tarea, documentarla y cerrarla en un solo paso.



4.1.2.e Diseño de flujos tratados como documentos

Como se ha explicado anteriormente, cada tarea será almacenada como entidad propia en la base de datos relacional y no será tratada como un documento. No obstante el diseño de los flujos sí será tratado como un documento en sí mismo, es decir la tarea de diseñar cada uno de los flujos que nuestra empresa cliente necesite consistirá en editar los documento del tipo "*flujo*". Como todos los documentos contendrá:

- **Atributos básicos de documento:** Identificador, tipo, fecha, etc.

- **Metadatos con la información del flujo:** Esto es, una descripción en formato JSON de las tareas, las acciones y los saltos que forman parte del flujo.
- **Procesos**: En este caso únicamente necesitaremos dos procesos:
  - **Editar el documento:** Para poder modificar,añadir o eliminar tareas y saltos de nuestro flujo de trabajo.
  - **Visualizar el documento:** En este caso, el hecho de abrir un documento de tipo "*flujo"* para visualizarlo consistirá en mostrar en la interfaz de usuario el formulario necesario para crear un nuevo flujo de trabajo de ese tipo. El formulario que se mostrará al usuario solicitará los datos básicos para crear la tarea padre y el documento de entrada de dicha tarea padre.

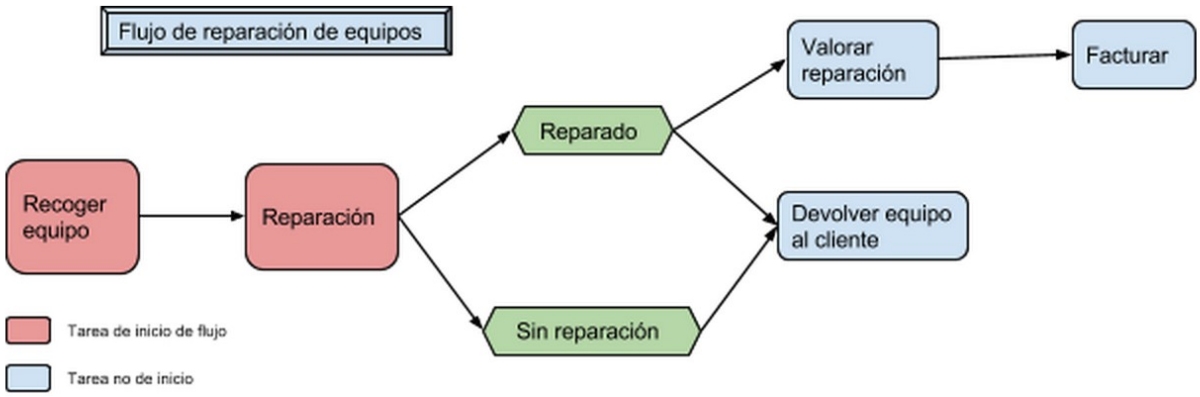
El resultado final es que se creará un documento de este tipo por cada flujo de trabajo indicado por el cliente. Los usuarios con privilegios de escritura sobre estos documentos serán los que podrán editar el flujo de trabajo, añadiendo tareas, saltos, etc. Los usuarios con privilegios de lectura sobre estos documentos serán los que podrán iniciar nuevos flujos de este tipo (en definitiva, iniciar un nuevo flujo es crear la tarea inicial del flujo y para poder hacerlo se necesita poder leer el flujo para averiguar cuál es su tarea de inicio).

Para que se entienda mejor, si estuviéramos hablando de recetas de cocina, quien tiene permisos para escribir en la receta puede añadir o quitar ingredientes, cambiar cantidades, etc y quien tiene permisos para leer la receta puede simplemente elaborar el plato.

4.1.2.f Múltiples tareas de entrada en el flujo

En algunas ocasiones podría darse el caso en el que dos flujos de trabajo a priori idénticos puedan inicializarse en dos tareas diferentes. En nuestro ejemplo anterior se podría dar el caso de necesitar otra tarea del tipo "Recogida del equipo" que consistiría en que en ciertas ocasiones el cliente no envía el equipo a nuestras instalaciones y es nuestro personal el que debe cumplir la tarea de ir a recogerlo. Una solución podría ser hacer una copia del documento que describe dicho flujo y modificarlo para que la tarea de inicio fuera diferente, con una descripción, un documento de entrada y un documento de salida distintos pero cuya subtareas fueran el mismo tipo de tarea que teníamos de inicio (Recepcionar equipo) y a partir de ahí los dos flujos fueran idénticos. Esta solución, aunque factible, podría suponer un exceso innecesario de trabajo a la hora de configurar nuestros flujos de trabajo. La solución final ha sido añadir un campo nuevo a nuestro documento de diseño de flujos de trabajo en el que podamos indicar cuáles de las tareas pueden ser las que inicien el flujo. Esto supone que a la hora de crear un flujo nuevo el usuario deberá seleccionar la tarea de inicio del mismo (en caso de existir varias).

En la siguiente figura se muestra el mismo ejemplo tratado anteriormente pero con varias tareas de inicio posibles. El cliente nos ha solicitado añadir la posibilidad de que antes de crear la orden de reparación haya que ir a recoger el equipo, es decir, en este caso será necesario crear una tarea “recoger equipo”, como en todas las tareas, un empleado deberá asignarse la tarea, una vez que la haya realizado y el equipo se encuentre en el taller podrá dar la tarea por cerrada, generándose de esta forma la siguiente tarea en el flujo, en este caso “reparación”. De esta forma, a la hora de crear un flujo de este tipo, el sistema deberá solicitar indicar la tarea inicial, es decir “Recoger equipo” si hay que ir a por él o “Reparación” si el equipo ya está en el taller.



4.1.3 Terceros

Será un componente que nos facilitará interrelacionar el resto de componentes y aportará información importante a una estructura de datos.

Un tercero podrá ser cualquier empresa, persona o entidad que tenga relación con nuestra empresa cliente. Por ejemplo:

- Empresas:
    - Clientes.
    - Proveedores.
    - Acreedores
    - Empresas asociadas al grupo.
- Personas:
    - Empleados.

    - Clientes.
    - Proveedores (Autónomos).
    - Usuarios del sistema.
    - Contactos o empleados de otras empresas.
- Entidades
    - Entidades bancarias
    - Servidores informáticos, robots, etc (A los que por ejemplo, les podremos asignar tareas).

Un tercero tendrá las siguientes relaciones con el resto de los componentes:

- Documentos: Cada documento podrá estar relacionado con un tercero (ya que “tercero” es uno de los atributos de un documento). Dependiendo del tipo de documento y del uso que queramos darle, esta relación podrá tener un significado pudiendo indicar que dicho tercero es el creador del documento, el destinatario, el remitente, etc. Esto no quita que un documento pueda tener más terceros asociados si estos aparecen referenciados en los metadatos del documento, por ejemplo, nuestro cliente podría querer guardar una lista de los empleados de cada uno de sus clientes, en este caso, una de las muchas posibilidades sería crear un documento de tipo “lista de empleados”, dicho documento tendría como atributo “tercero” el ID de la empresa cliente y en sus metadatos guardar un listado de los IDs correspondientes a los empleados (que serán tratados como terceros también).
- Tareas. Tendrá una doble relación:
    - Tercero que crea o solicita la tarea.
    - Tercero al que se le asigna la tarea y la ejecuta.
- Contabilidad: Cada una de las cuentas contables de nuestro libro de cuentas (y por consiguiente, cada uno de los asientos contables introducidos en el sistema) podrán estar relacionados con un tercero.

4.1.4 Libro de cuentas

Será un componente típico de gestión contable para empresas. No será muy diferente al resto de herramientas ya existentes en el mercado ya que el plan general de contabilidad es muy concreto, la única particularidad en nuestro caso será que se integrará perfectamente con el resto del sistema pudiendo relacionar cada una de las cuentas y cada uno de los asientos con el resto de componentes anteriores (documentos,

tareas y terceros).

4.1.4.a Introducción a la contabilidad

Una breve introducción a la contabilidad, sin entrar en detalle, únicamente para clarificar conceptos que necesitaremos tener claros para entender el funcionamiento de nuestro sistema:

De manera un poco burda, podríamos decir que la contabilidad es una metodología que nos permite:

- Por un lado cuantificar y clasificar los activos de una empresa (para lo que haremos usos de las cuentas contables).
- Registrar las transacciones o movimientos de los activos de una cuenta contable a otra (para lo que haremos uso de los asientos contables).

Cuentas contables

El plan general de contabilidad recoge todos los tipos de cuentas contables entre los que puede haber un flujo de activos y los divide en siete grupos:

- Grupo 1: Financiación básica.
- Grupo 2: Inmovilizado.
- Grupo 3: Existencias.
- Grupo 4: Acreedores y deudores.
- Grupo 5: Cuentas financieras.
- Grupo 6: Compras y gastos.
- Grupo 7: Ventas e ingresos.

Por cada uno de estos grupos existe una serie de tipos de cuentas contables identificadas por un número. Por ejemplo, algunos de los más habituales:

Grupo	Tipo	Nombre
1	10	Capital social
4	40	Proveedores
4	41	Acreedores
4	43	Clientes
4	46	Personal
4	47	Administraciones públicas

5	57	Tesorería
6	60	Compras
6	62	Servicios
7	70	Ventas

El P.G.C. divide a su vez la mayoría de estos tipos en varios subtipos, dependiendo del nivel de detalle que se quiera tener en el control de la contabilidad se puede hacer uso de esta subdivisión o no. Por ejemplo, lo normal es distinguir en las cuentas de tipo 57 (tesorería) las que son cuentas de caja (tipo 570) y las que son cuentas bancarias (tipo 572). Y es casi imprescindible separar las cuentas del tipo 47 (administraciones públicas) teniendo: 470 para “IVA soportado”, 477 para “IVA repercutido”, 476 para “Seguridad social”, etc.

Por cada uno de estos tipos de cuentas contables, lo habitual será tener varias cuentas, cada una de ellas la identificaremos por el tipo de cuenta seguido de un identificador numérico dentro de ese tipo. Por ejemplo, del grupo 43 (clientes) crearemos una cuenta por cada uno de nuestros clientes. 43-0001=cliente1, 43-0002=cliente2, etc. Para el grupo de 40 (proveedores) crearemos una cuenta por cada uno de nuestros proveedores, 40-0001=proveedor1, 40-0002=proveedor2, etc.

Un libro de cuentas habitual en una empresa quedaría de la siguiente manera:

Cuenta	Nombre
40-0001	Proveedor1
40-0002	Proveedor2
40-0003	Proveedor3
41-0001	Acreedor1
41-0002	Acreedor2
41-0003	Acreedor3
43-0001	Cliente1
43-0002	Cliente2
43-0003	Cliente3
465-0001	Empleado1
465-0002	Empleado2
465-0003	Empleado3
472-0000	IVA soportado
475-0000	Hacienda
476-0000	Seguridad social

477-0000	IVA repercutido
570-0001	Caja
572-0001	Cuenta bancaria 1
572-0002	Cuenta bancaria 2
600-0001	Compra de repuestos
602-0001	Gastos oficina
610-0001	Alquiler local 1
624-0001	Gasolina
624-0002	Servicios de mensajería
626-0001	Comisiones bancos
628-0001	Agua
628-0002	Luz
629-0001	Teléfono
642-0001	S.S. a cargo de la empresa
700-0001	Ventas producto A
700-0002	Ventas producto B

Asientos

Se llama “asientos” a los movimientos contables por los cuales se traspasa un valor de una cuenta a otra. Por cada asiento deben intervenir al menos dos cuentas contables (una de la que se descuenta el valor y otra a la que llega dicho valor) pero pueden ser muchas más. Es imprescindible que la suma de los valores sustraídos de las cuentas contables y los valores sumados sea 0. Lo habitual en contabilidad es representar los asientos con dos columnas diferentes para diferenciar los valores que se descuentan de una cuenta contable (DEBE) y los valores que se suman a una cuenta contable (HABER), por lo explicado antes, la suma del DEBE debe ser igual a la suma del HABER. Algunos ejemplos comunes de asientos contables:

Asiento factura emitida:

Cuenta	Concepto	DEBE	HABER
70-0001	Venta		100€
477-0001	IVA repercutido		21€
430-0001	Factura 123/2015	121€	

Es decir, este asiento de venta consta de 3 apuntes contables, declaramos 100€ de

beneficio por una venta, 21€ irán a la cuenta de IVA repercutido para cuando debamos declarar impuestos y 121€ a la cuenta de nuestro cliente ya que pasa a ser una deuda que este tiene con nosotros.

Asiento de pago de factura(de un cliente a nosotros)

Cuenta	Concepto	DEBE	HABER
572-0001	Pago por transferencia	121€	
430-0001	Pago Factura 123/2015		121€

En este asiento de pago únicamente aparecen 2 apuntes. Por un lado el sumamos los 121€ que han llegado a nuestra cuenta bancaria y por otro lado restamos la misma cantidad de la cuenta del cliente que rebaja así su deuda con nosotros.

Balance de las cuentas

Consiste en sumar todos los DEBE y todos los HABER de una cuenta para conocer el saldo actual de una cuenta contable. Por ejemplo, para los ejemplos anteriores, el saldo de nuestro cliente 43-0001 sería:

Cliente 43-0001	Concepto	DEBE	HABER
	Factura 123/2015	121€	
	Pago Factura 123/2015		121€
	Total	121€ - 121€ = 0€	

Esto significaría que el cliente ha pagado todas las facturas que se han emitido a su nombre y por lo tanto sus saldo es 0€

4.1.4.b Componente de contabilidad

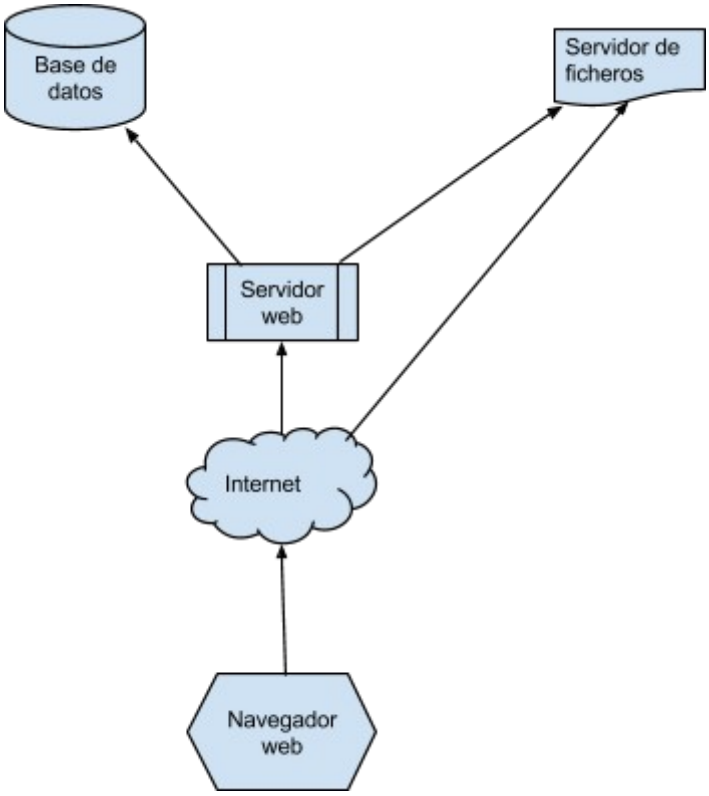
Este componente se encargará principalmente de gestionar:

- Cuentas contables.
    - Nombre de la cuenta.
    - Grupo y subgrupo al que pertenece en el PGC.
    - Identificador único (Número de cuenta).
    - Tercero asociado a la cuenta.
- Asientos contables.
    - Identificador único (Nº de asiento).

- Fecha contable.
- Fecha valor.
- N* apuntes:
    - Cuenta.
    - Concepto
    - Valor

4.2 Modelo de implementación

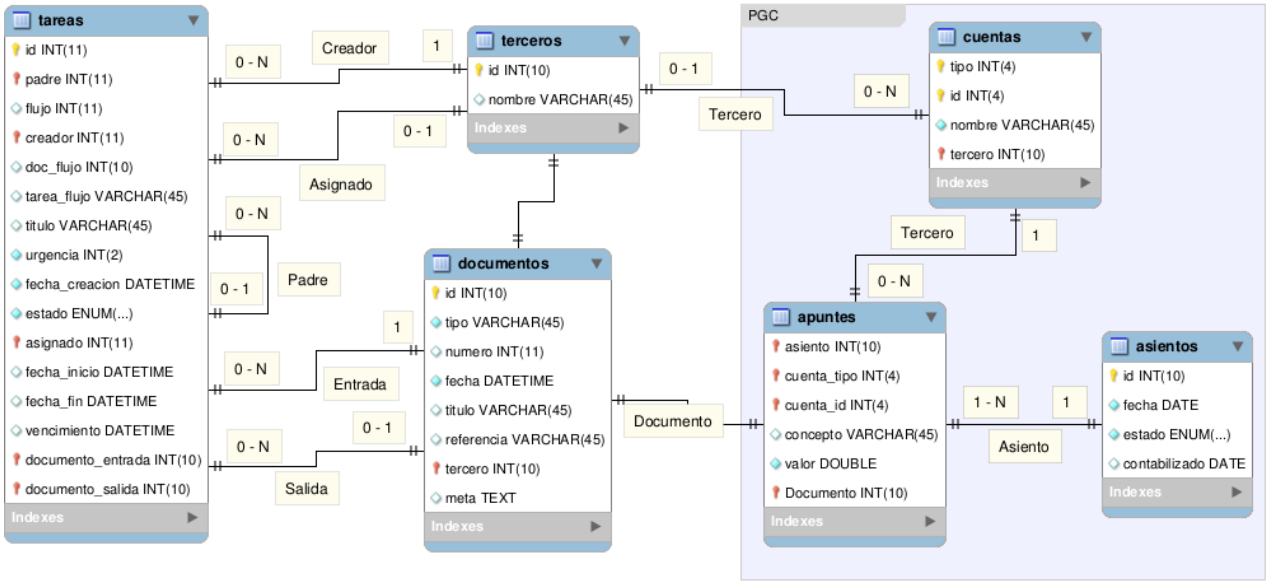
- **Base de datos relacional:** Que nos permitirá almacenar la información de nuestros componentes lógicos.
- Servidor web: Que nos permitirá interactuar con nuestros componentes lógicos.
- Servidor de ficheros: Básicamente, nos permitirá almacenar ficheros que queramos o necesitemos adjuntar a nuestros documentos pero que por rendimiento o capacidad no queremos almacenar en la base de datos relacional. Nuestro cliente podrá plantearnos diferentes formas de acceder a estos ficheros y aunque en primera instancia haremos que nuestra interfaz web sea capaz de gestionar dichos ficheros (siendo en este caso el servidor de ficheros un sistema de ficheros accesible desde el propio servidor web) dejamos abierta la posibilidad de delegar esta función en otro tipo de servidores que podrán ser propios de la empresa (Servidor SAMBA, FTP, etc) o de servicios externos a esta (Dropbox, Google Drive, etc).



4.2.1 Base de datos relacional

El diseño final de nuestra base de datos cumple nuestros requisitos pre-establecidos de simplicidad pero no por ello es menos potente.

Nuestro Diagrama entidad relación sería el siguiente:



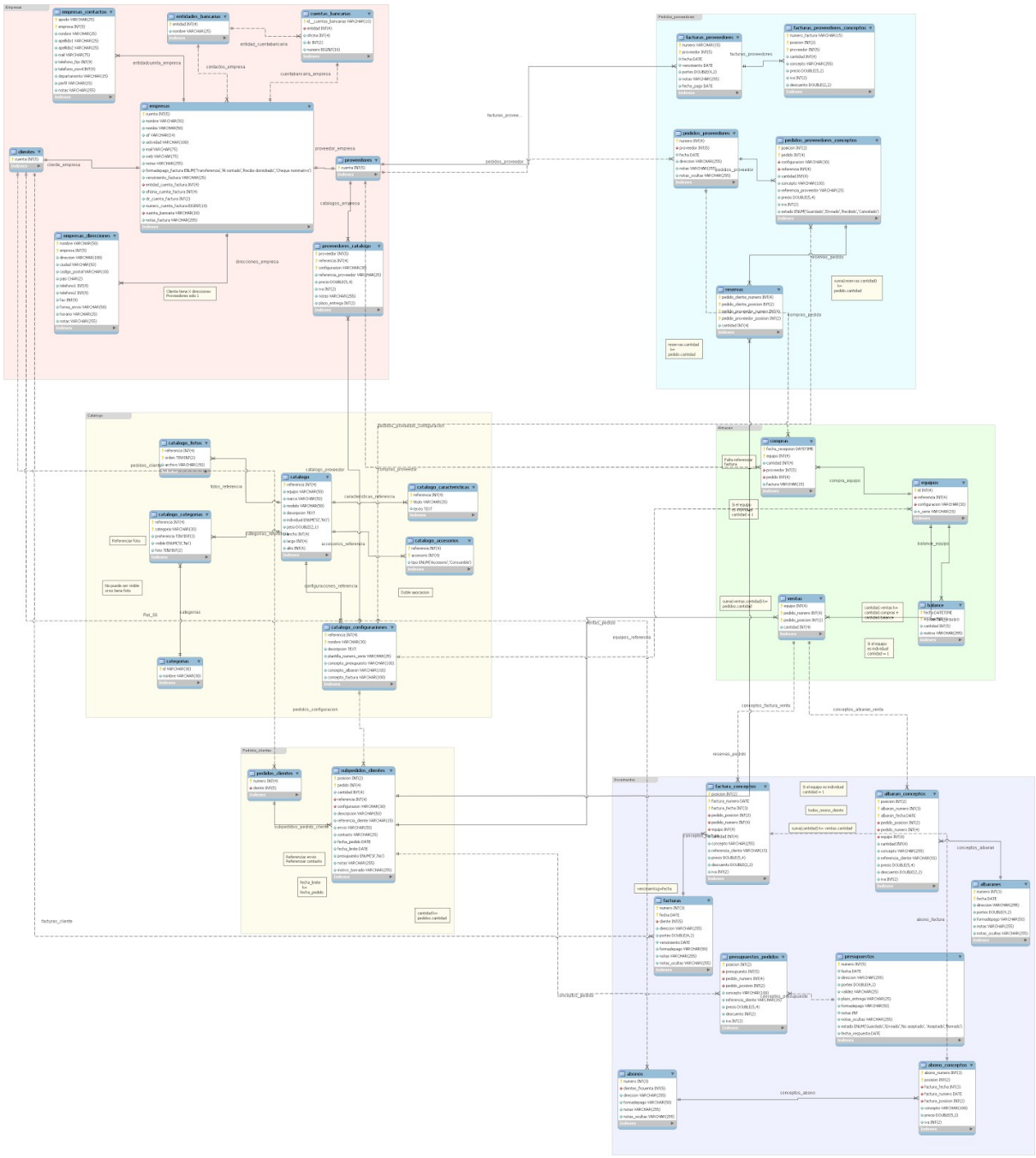
Como el resto de nuestro sistema, este modelo entidad relación será extensible para adaptarse a futuros requisitos de nuevos clientes.

4.2.1.a JSON

Los gestores de bases de datos más actuales permiten contener campos de tipo JSON y hacer querys en base a la información contenida en ellos. Los principales sistemas gestores de bases de datos que permiten esto son:

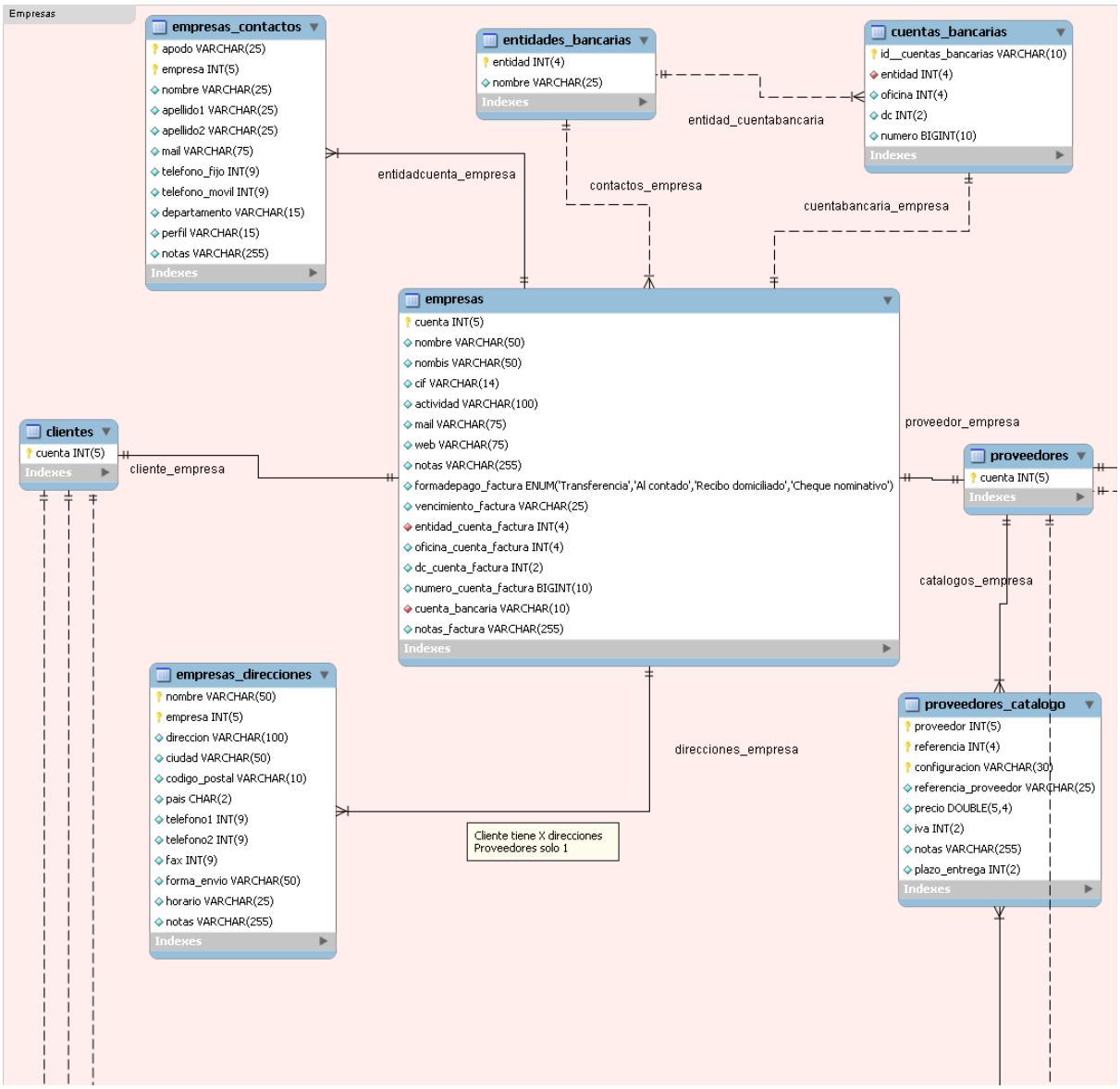
- MySQL 5.7 (23/4/2013)
- MariaDB 10.0.1 (6/2/2013)
- Oracle 12.1.0.2 (22/7/2014)
- PostgreSQL 9.2 (10/09/2012)

A modo comparativo, el siguiente sería un modelo EER típico para una empresa dedicada a la compra-venta y que necesita manejar clientes, proveedores, catálogos, pedidos, facturas, albaranes, etc. Este modelo EER sería perfectamente sustituible por nuestra base de datos relacional.



¿Sería posible almacenar y organizar la misma información en nuestro modelo de datos?

Por ejemplo, tomamos la parte que hace referencia a la gestión de empresas:



En primer lugar, por cada registro de la tabla “Empresas” crearíamos un registro de nuestra tabla “Terceros” y por cada uno de ellos necesitamos un documento de tipo “datos empresa” para almacenar sus atributos (NIF, actividad, etc) en formato JSON dentro de los metadatos:

```
{
"Nombre": "Universidad Carlos III de Madrid",
"CIF": "A12345678",
"Actividad": "Universidad pública",
....
"Notas factura": "Enviar 3 copias"
}
```

Por cada registros en “empresa_contactos” crearíamos un nuevo documento de tipo “contacto empresa”, con todos los atributos necesarios en sus metadatos y relacionado con el tercero correspondiente a la empresa.

Podríamos actuar del mismo modo y crear documentos diferentes para sustituir las tablas “cuentas_bancarias”, “empresas direcciones” y “catálogo”, pero también tenemos la posibilidad de incluir esta información ampliando los metadatos del documento.

```
{
   "Nombre": "Universidad Carlos III de Madrid",
   "CIF": "A12345678",
   "Actividad": "Universidad pública",
   ....
   "Notas factura": "Enviar 3 copias",
   "Cuentas bancarias": [
      "cuenta1": "12345678901234567890",
      "cuenta1": "24680135792468013579",
   ],
   "Direcciones": [
      "direccion1": {
         "Dirección": "Av. Universidad 30",
         "Ciudad": "Leganes",
         ...
      },
      "direccion2": {
         "Dirección": "c./ Madrid 126",
         "Ciudad": "Getafe",
         ...
      },
   ]
}
```

En este caso, nosotros clasificamos los terceros como clientes o proveedores creando asignándoles las cuentas del grupo correspondiente en el plan general de contabilidad, a los terceros que sean clientes les asignaremos una cuenta del grupo 430 y a los proveedores una cuenta del grupo 410. Por tanto las tablas “clientes” y “proveedores” desaparecen.

4.2.1.b Vistas prediseñadas en la base de datos

Para facilitar el desarrollo de la aplicación, enriqueceremos la base de datos con una serie de “views” o querys prediseñadas que nos faciliten obtener datos de la base de datos. Estas vistas nos permitirán obtener datos cruzados de la base de datos. Por ejemplo:

- view_diario: Permitirá obtener en un sólo paso el diario de una cuenta, para ello cruzará datos entre la tabla “apuntes” y “asientos”.
- view_terceros: Permitirá obtener en un sólo paso información ampliada de un tercero. Cruzando los datos con la tabla “cuentas” para conocer el tipo de tercero que estamos tratando y el balance total del tercero.
- view_documentos: Permite ampliar la información de los documento, cruzando por ejemplo la información con la tabla “terceros” para obtener directamente el nombre del tercero asignado al documento.

4.2.2 Interfaz web

Será un componente imprescindible para nuestro sistema ya que será el que permitirá interactuar al usuario con la información contenida en nuestra base de datos. No obstante, como ya se ha explicado anteriormente, el principal contenedor de información de nuestro sistema será nuestro componente “Documento” y en él recae además la mayor parte de la lógica de nuestras empresas cliente y por lo tanto la única función de nuestra interfaz web será mostrar y hacer accesible dichos documentos.

Ya que nuestra interfaz web será previsiblemente uno de los componentes que más necesitaremos modificar y adaptar a nuestros clientes (ya que cada uno tendrá sus gustos), hemos tratado de hacerlo igualmente versátil, esto significa que nuestra interfaz web y su composición también lo trataremos como un conjunto de documentos. Para conseguirlo, necesitamos crear una serie de documentos que llamaremos “documentos del sistema”. Estos “documentos del sistema” se tratarán exactamente igual que el resto de documentos, contendrán los mismo atributos y podremos realizar las mismas operaciones de editar, abrir, etc.

Para nuestra interfaz web inicial únicamente hemos necesitado diseñar dos tipos de “documentos del sistema”, pero obviamente, podremos añadir más en el futuro. Estos dos primeros diseños son:

- **Menús:** Podremos editar este documento para añadir enlaces a cualquier otro punto de nuestra interfaz (Documentos, otros menús, listas, etc). Al tratar cada menú como un documento más de nuestro sistema podremos aprovechar las mismas funcionalidades, por ejemplo:

- Asignar un menú diferente a cada uno de nuestros usuarios (terceros).
- Copiar un menú tantas veces como sea necesario para utilizarlo como base y añadir funcionalidades posteriormente.
- Podremos tener tantos menús como sean necesarios y poder vincularlos entre ellos.
- etc.

- **Listas:** Es otro elemento que hemos necesitado implementar para hacer completamente funcional nuestra interfaz web. Al editar este tipo de documentos podremos diseñar una tabla donde mostrar la información contenida en nuestra base de datos.
La forma más sencilla de diseñar dicha tabla consistirá en seleccionar la fuente de la información que queremos mostrar de un listado de "vistas" prefijado en el sistema y a continuación indicar cada uno de los campos que deseamos hacer visibles en el listado y bajo qué condiciones se mostrarán. Será casi como diseñar visualmente una Query a nuestra base de datos. El listado de vistas disponibles incluirá cada una de las entidades que forman parte de nuestro modelo de datos (Documentos, terceros, tareas o asientos) pero también podrá incluir vistas que hagan referencia a Querys más complejas en la base de datos (JOINs de tablas, transformaciones de campos, etc). Además, existirá una última opción en la fuente de datos llamada "Otro" que permitirá al usuario introducir manualmente una Query en formato SQL, obviamente, seleccionando esta última opción el sistema no será capaz de ofrecernos los campos disponibles por lo que el usuario deberá introducirlos también a mano.

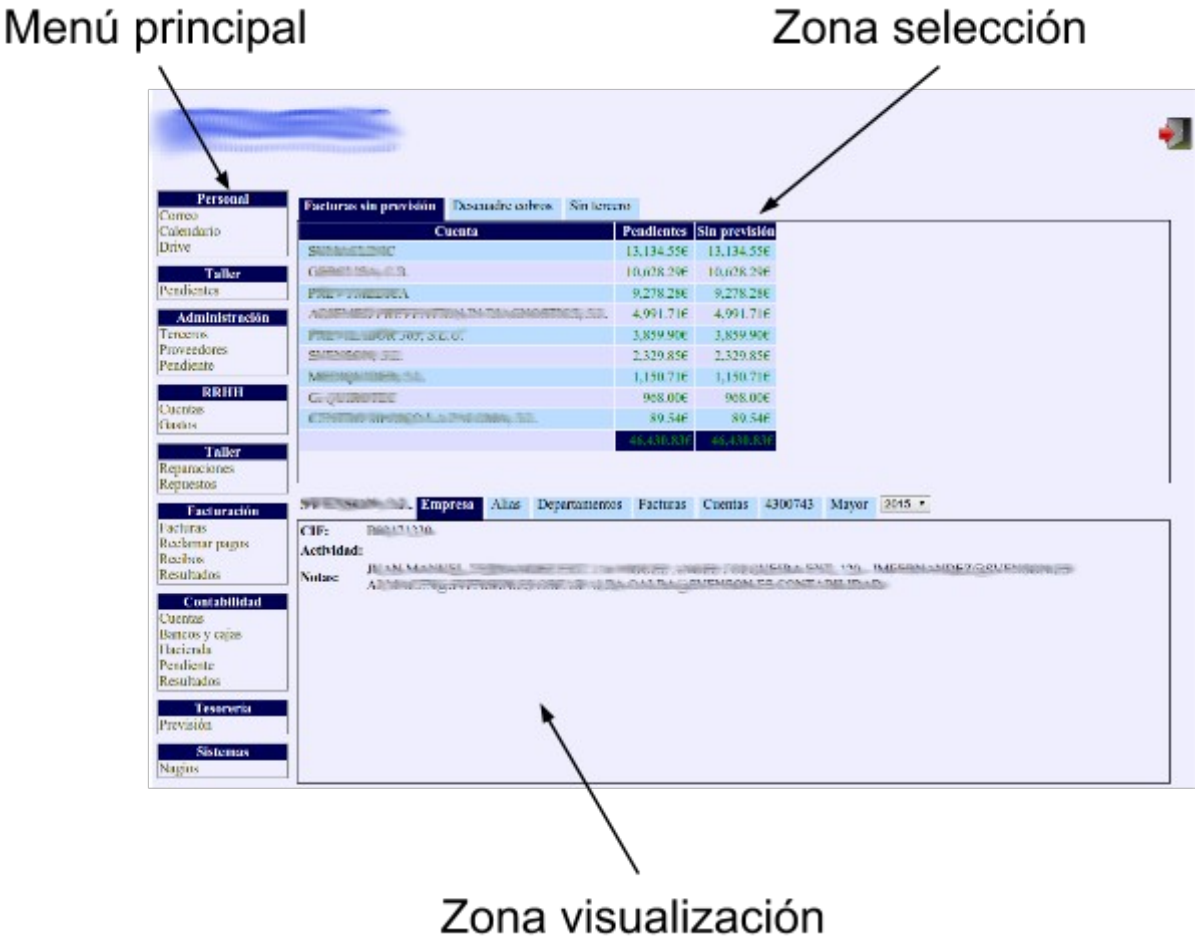
4.2.2.a Interfaz preliminar

A falta de conocer los requisitos finales de nuestros posibles clientes, se ha diseñado una interfaz que nos permita probar la funcionalidad de nuestros componentes.

Hemos dividido esta interfaz en tres zonas diferentes:

- Menú principal. Como hemos explicado antes, podremos diseñar un menú diferente para cada usuario según sus roles y necesidades. Este menú nos servirá para desplegar la información que necesitemos en cada momento en cualquiera de las otras dos zonas de nuestra interfaz:
- Zona de selección. Destinada a mostrar otros submenus y listas prediseñadas que nos permitan buscar y seleccionar el dato o la información que necesitamos.

- Zona de visualización: Dependiendo del tipo de elemento seleccionado en la zona de selección mostraremos de forma ordenada toda la información. Tendremos



4.2.2.b Desarrollo de documentos

A la hora de crear un nuevo tipo de documento existen una serie de restricciones que debemos de tener en cuenta:

La localización de todos los ficheros necesarios para el funcionamiento de los documentos deberán estar dentro del directorio “~/formularios/$PATH$/”, siendo $PATH$ el nombre con el que daremos de alta el documento en el sistema.

Al invocar un documento, el sistema llamará al fichero “~/formularios/ $PATH$/index.inc”, el cual debe ser programado para responder correctamente a las solicitudes que le haga sistema. Aunque para documentos sencillos no será necesario crear más ficheros que el “index.inc” se recomienda que para otros más complejos se separe en otros ficheros por funcionalidades (editar, visualizar, imprimir, etc).

El fichero “index.inc” deberá actuar en función de una serie de argumentos que el

sistema le enviará dependiendo de la acción del usuario sobre la interfaz web:

- $documento: Esta variable contiene un Array con todos los atributos del documento al que se desea acceder. Si no existiese esa variable, se interpretará que se trata de un documento nuevo.
- $_REQUEST['ids']: Esta variable contiene un array de IDs de documentos, esto significa que el usuario ha decidido crear un documento nuevo a partir de los documentos ya existentes a los que hace referencia el array.
- $modo_apertura_documento: Esta variable de tipo texto tendrá los siguientes valores dependiendo del modo de apertura del documento:
    - “visualizacion”: El documento deberá visualizar los datos pero no permitirá modificarlos. Este modo será el modo por defecto en el caso de indicar un $modo_apertura_documento no reconocido o programado.
    - “edicion”: En este modo el documento debe mostrarse de tal forma que permita su edición y guardado, es decir, cada uno de los campos se convertirá en un elemento de formulario y se deberá mostrar un botón de “guardar”.
    - “imprimir”: En este modo el documento debe mostrar los datos de tal forma que estén preparados para ser impresos. Si fuera necesario habrá que incluir elementos como cabeceras, numeración de páginas, etc que no aparecen en el modo “visualización”.
    - “descargar”: En este modo el documento debe entender que el usuario desea guardar una copia de los datos en su dispositivo. Dependiendo del tipo de documento se creará un fichero del tipo necesario, puede ser un PDF, un ZIP, etc.

Otro fichero imprescindible dentro del directorio “~/formularios/$PATH$/” será “save.php”. El sistema llamará a este script cada vez que se deba guardar el documento y lo hará con una serie de argumentos que se deberán interpretar para el correcto funcionamiento:

- $_REQUEST['form_doc_id']: Será el ID del documento que vamos a guardar. En caso de no indicarse dicho ID se interpretará que se trata de un documento nuevo y por tanto se le asignará un ID nuevo.
- $_REQUEST['$NAME$']: Donde $NAME$ será el nombre que le hemos dado a cada uno de los elementos del formulario que se han creado al abrirlo en modo edición.

A la hora de programar el documento habrá que tener especial cuidado de que el

formato usado por el script “save.php” para guardar los datos sea compatible con el formato usado por el script “index.inc” para recuperar esos datos.

A la hora de dar de alta nuestro documento creado será tan sencillo como crear un nuevo documento de tipo “Definición de documento” donde deberemos indicar el título que queremos darle y el $PATH$ que hemos usado anteriormente.

4.2.2.c Botones

A la hora de diseñar el modo “edición” de los documentos se podrán incluir todas las etiquetas HTML5 que se necesiten (siempre y cuando sean compatibles con el navegador web utilizado por el cliente) y programar cualquier modificación o personalización sobre estas etiquetas que faciliten al usuario su uso (autocompletado, validación de contenido, etc).

Previsiblemente tendremos que reutilizar en más de una ocasión alguno de estos elementos personalizados. Por ejemplo, seguramente necesitaremos en más de una ocasión un botón que permita al usuario seleccionar rápidamente a alguno de los clientes introducidos previamente en el sistema, para ello crearemos un botón que genere una lista de todos los terceros del sistema que estén asociados a una cuenta contable de cliente.

Todos los botones de este tipo que queramos poder reutilizar en más de un documento los guardaremos en el directorio “~/botones/$NOMBRE$” siendo $NOMBRE$ el identificador para poder ser llamado desde cualquiera de los documentos.

4.2.3 Sistema de ficheros

Como ya explicamos anteriormente, la mayor parte de la información de nuestro sistema estará contenida en los documentos, pero habrá ocasiones en los que necesitemos guardar ficheros en bruto, sobre todo cuando nuestra empresa cliente utilice algún otro software para su labor comercial. En estos casos, los metadatos en formato JSON de nuestros documentos nos permitirán crear enlaces a otros sistemas de ficheros que contengan estos archivos (servidores samba, webdav, google docs, dropbox o similares). Esto nos permitirá por ejemplo:

- Guardar en nuestro sistema cualquier tipo de formato de archivo, fotografías, PDFs, etc.
- En el caso de un estudio de arquitectura subir un fichero creado con el software Autodesk Autocad cuando el documento de salida requerido para una tarea sea la creación de ese fichero de tipo CAD.

4.3 Solución de requisitos en nuestro sistema

Como hemos visto, todos los requisitos referentes a la contabilidad y a la gestión de tareas les hemos dado solución a base de diseñar un componente específico para ello. El resto de requisitos se ha solucionado con nuestro diseño de componentes lógicos:

Requisito	**Solución**

Gestión de clientes y proveedores	Serán terceros con:
	• Cuentas contables del grupo 4 asociadas
	• Documentos asociados con el que trataremos su información del tipo:
	  • Datos de empresa.
	  • Datos de facturación
	  • Departamentos.
	  • Contactos.
	  • Direcciones
	  • Etc.

Gestión de empleados	Serán terceros con:
	• Cuentas contables asociadas
	• Documentos del tipo:
	  • Nómina
	  • Datos personales.
	  • Direcciones.

Gestión de catálogo	Serán documentos: • Catálogo. • Productos. • Fotos y descripciones
Gestión de tarifas	Serán documentos de: • Tarifa: • Productos
Gestión de almacenes	Serán documentos de: • Datos de almacén. • Albaranes de entrada. • Albaranes de salida. • Inventarios • Direcciones

Gestión de usuarios	Cada usuario será un tercero (previsiblemente un tercero que también estará asociado a documentos de empleado). Para su gestión se le asociarán documentos de: • Datos de autenticación en el sistema • Roles de usuario.

4.4 Infraestructura necesaria

4.4.1 Servidor de base de datos

Todas las operaciones de nuestra aplicación sobre base de datos están agrupadas en un mismo fichero para facilitar que nuestro software sea fácilmente migrable a cualquier otro sistemas de base de datos. Por lo tanto nuestro sistema es compatible con cualquier sistema gestor de bases de datos relacional. Preferiblemente compatible con JSON para permitir al desarrollador agilizar algunas operaciones con los metadatos de los documentos.

Los SGBD recomendados son:

- MySQL 5.7 (23/4/2013)
- MariaDB 10.0.1 (6/2/2013)
- Oracle 12.1.0.2 (22/7/2014)
- PostgreSQL 9.2 (10/09/2012)

En cuanto a los requisitos mínimos del servidor dependerán (a parte de los requisitos recomendados por el propio SGBD) de nuestra empresa cliente:

- Número de transacciones.
- Número de usuarios conectados al mismo tiempo.
- Formato de los documentos.
- ¿Necesita alta disponibilidad de todos sus sistemas?

Para cada empresa cliente a la que adaptemos nuestro sistema necesitaremos diseñar la arquitectura de sistemas necesaria y realizar una serie de pruebas de carga que nos aseguren el correcto funcionamiento del mismo.

4.4.2 Servidor web

La interfaz web está totalmente escrita en HTML y PHP por lo tanto será necesario un servidor HTTP capaz de interpretar código PHP. Para las pruebas hemos usado un servidor Apache 2.2.

Al igual que con la base de datos, los requisitos mínimos dependerán igualmente de los requisitos recomendados para el servidor web elegido y del volumen de la empresa cliente.

4.4.3 Servidor de ficheros

Nuestra estructura de metadatos en formato JSON nos permite hacer referencia a cualquier tipo de localización de ficheros, ya sea propio o de un tercero. Para las pruebas hemos utilizado el propio servidor web como repositorio de ficheros pero es perfectamente compatible con servidores Samba, WebDav, Google Docs, Dropbox, etc.

4.4.4 Cliente

Para acceder a nuestra interfaz web necesitaremos un navegador web compatible 100% con HTML5. Dependiendo del navegador utilizado, si no cumple al 100% con el estándar HTML5 es posible que algunas funcionalidades no estén disponibles.

4.5 Posibles problemas y mejoras

Rendimiento

La generalización de nuestros componentes tiene un importante contra y es la pérdida de rendimiento en cuanto a indexación, búsquedas, ordenación etc. Para bases de datos de pequeñas empresas no debería ser un problema pero sí en el caso de una gran empresa. Para resolverlo deberemos acudir a técnicas de escalado y cacheado en servidores, las cuales son muchas y algunas muy potentes que acelerarán considerablemente nuestro sistema. A priori, se considera que las primeras acciones para aumentar el rendimiento de nuestro sistema podrían ser:

- Cambiar el SGBD MySQL por otro más potente como PostgreSQL.
- Usar sistemas de cache en RAM como Memcached.
- Escalar nuestros servidores para repartir la carga entre varios servidores formando clusters de bases de datos o de los servidores web.
- Aumentar el rendimiento del almacenamiento con sistemas RAID profesionales o discos de estado sólido (SSD).

- Etc

Adaptación de los empresarios a la nube.

Como hemos comentado anteriormente, nuestro sistema está pensado para funcionar en servidores y no en PCs, de tal forma que podremos ofrecerle a nuestros clientes alojar la aplicación en sus propios servidores, no obstante nuestro producto está pensado para PYMES y micro PYMES y por lo tanto es muy posible que no dispongan de la infraestructura necesaria, en ese caso podremos ofrecerle al cliente la posibilidad de alojar la aplicación en servidores externos pero esto implica la mentalización por parte de los empresarios de que los datos de su empresa no estarán almacenados en el disco duro de su PC sino en una nube de servidores.

API

Como ya hemos explicado, nuestro sistema está pensado para que no sólo sean personas físicas las que interactúen con él sino también otras entidades, servidores o robots que puedan realizar tareas para nuestra empresa cliente. Para facilitar este trabajo sería conveniente diseñar una API que facilite las tareas de loguearse en nuestro sistema, listar, tareas, capturarlas, etc.

Seguridad

Al tratarse de datos tan confidenciales como puede ser la contabilidad de nuestra empresa y al tratarse de un sistema diseñado para la nube, la seguridad será una tarea de mejora continua. Aunque durante el desarrollo se han efectuado pruebas que demuestran que nuestro sistema es seguro, en este apartado de posible mejoras debemos incluir el diseño de una batería de pruebas de seguridad que podamos ir enriqueciendo a medida que añadimos funcionalidades. Esta batería de pruebas la tendríamos a nuestra disposición para comprobar que después de hacer modificaciones en cualquier parte del código nuestro sistema sigue siendo seguro.

integración con otras herramientas

Para facilitar la migración de nuestro futuros posibles clientes a nuestro sistema sería necesario que este fuera compatible con el mayor número de herramientas de las más utilizadas por el resto de empresas. Por eso sería interesante diseñar planes de migración desde estas otras herramientas que nuestro sistema pretende sustituir.

Lector correo

En nuestro afán de brindar una solución global para la gestión de toda la empresa a nuestros clientes no debemos olvidar que el correo electrónico es la herramienta principal

de trabajo para la mayoría de empresas, por eso se incluye en esta lista de posibles mejoras la posibilidad de convertir nuestro sistema en un lector de correo electrónico. A falta de un diseño más exhaustivo, la idea sería brindar al sistema de la capacidad de leer nuestra cuenta de correo electrónico y que este convirtiera todos nuestros correos en tareas y todos los adjuntos en documentos.

Valoración de tareas

Una mejora muy interesante para nuestro sistema sería ofrecer al usuario la opción de valorar el trabajo y el coste que ha supuesto cada una de las tareas que ha realizado pudiendo indicar por ejemplo, el tiempo que ha necesitado para su realización o los recursos que ha necesitado emplear.

Esto permitiría a nuestra empresa realizar estudios para aprovechar mejor los recursos disponibles y mejorar su rentabilidad.

Diagramas Gantt

Ya que en nuestro sistema podremos diseñar flujos de tareas, asignar dichas tareas a nuestros usuarios y estimar el tiempo y los recursos necesarios, sería muy interesante tener la capacidad de crear diagramas Gantt a partir de toda esta información.

De esta forma ampliaremos nuestro abanico de clientes objetivo pudiendo añadir empresas dedicadas a la gestión de proyectos.

Idiomas y divisas

Otra forma de ampliar nuestro abanico de posibles clientes y usuarios sería añadir nuevos idiomas y la posibilidad de seleccionar diferentes divisas.

Versionado de documentos

Al igual que ya hacen otros gestores documentales existentes en el mercado podremos ampliar las características de nuestro sistema añadiendo un control de versionado de los documentos que permita al usuario revisar los cambios realizados a lo largo del tiempo en sus documentos y en caso de ser necesario volver a una versión anterior.

Calendarios

Nuestro sistema ya cuenta con toda la información necesaria para poder ofrecer a cada usuario un calendario que le muestre todas las tareas que tiene pendientes de realizar. La idea sería ofrecer en la interfaz un calendario que permitiera al usuario visualizar todas las tareas que tengan una fecha de vencimiento fijada.

Permisos en los documentos

Ya que nuestro sistema puede funcionar también como un sistema de ficheros, sería interesante añadirle todas las posibilidades de los sistemas de ficheros tradicionales. De esta forma podríamos asignarle a cada documento no sólo un propietario sino también un grupo o rol y ofrecer al usuario la posibilidad de asignar permisos de lectura y/o escritura dependiendo del usuario que accede y los grupos a los que pertenece, de tal forma que los usuarios que pertenezcan al mismo grupo o rol hereden esos permisos sobre el documento.

Gestión de múltiples empresas

Pensando en que uno de nuestros potenciales clientes serían las empresas dedicadas a la gestoría de empresas será necesario ofrecerles una forma rápida de poder acceder con un mismo usuario a la información contable de todas las empresas que dicha gestora tenga como clientes.

4.6 Tipos de documentos creados

Como ya se ha mencionado anteriormente, para el correcto funcionamiento del sistema se han diseñado una serie de documentos que nos facilitarán su configuración.

4.6.1 Documentos del sistema

4.6.1.a Definición de documento

Será el tipo de documento que debamos crear para dar de alta el resto de documentos que diseñemos para nuestro sistema.

A la hora de editar este tipo de documento deberemos indicar el título con el que se mostrará en el sistema y el path que indica la ruta a los scripts “index.inc” y “save.php”.

El modo de visualización de los documentos de este tipo será la interfaz que nos permita crear un nuevo documento del tipo que definen.

En el modo de “download” de estos documentos permitirá descargar en un fichero zip toda la información de ese tipo de documentos y todos sus scripts de forma que podamos añadir este tipo de documentos en las interfaces web de otros clientes.

4.6.1.b Menú

Será el tipo de documento que nos permita diseñar menús con los que navegar por nuestro sistema.

A la hora de editar este tipo de documentos podremos crear una lista de enlaces a

otros menús, listas, etc.

En el modo de visualización de estos documentos se visualizarán los enlaces para que el usuario pueda seleccionarlos y acceder a los datos que necesita.

4.6.1.c Lista

Este tipo de documentos permitirá realizar listados que contengan información concreta y ordenada de algún tipo.

En el modo de edición de estos documentos se le ofrecerá al usuario la opción de seleccionar el tipo de datos que quiere en la lista (documentos, tareas, terceros,etc) y los campos concretos que quiere que aparezcan y bajo qué condiciones deben mostrarse o no en dicho listado.

En el modo de visualización, el documento generará la Query necesaria para obtener los datos indicados de la base de datos y mostrará una tabla HTML con dichos datos.

Los datos de este documento son:

- Título de la lista.
- Origen de los datos: Tabla o vista de la base de datos.
- Campos del listado.
    - Campo a mostrar.
    - Tipo del campo: Para decidir como se mostrará en el listado.
    - Filtro: Permite decidir que registros se mostrarán en el listado y cuales no.
    - Formulario: Para indicar que el campo será seleccionable desde el formulario de filtros.
- Opciones: Cuando el listado sea seleccionable (ver manual de usuario).
    - Opción: Título de la opción.
    - Documento: Documento que se generará con dicha opción.

4.6.1.d Flujo

Es el tipo de documentos que permitirá al usuario diseñar los diferentes flujos de trabajo que requiera la empresa.

En el modo de edición el usuario podrá crear tantas tareas, acciones y saltos

necesarios para el correcto funcionamiento del flujo.

En el modo de visualización, el documento mostrará el formulario necesario par crear una nueva tarea que inicie un flujo de este tipo.

Los datos de este documento son:

- Tareas:
    - Tarea: Título de la tarea dentro del flujo.
    - Documento de entrada.
    - Documento de salida.
    - Inicial: Para seleccionar que el flujo se podrá iniciar por dicha tarea.
- Acciones:
    - Título: Nombre de la acción que se mostrará en la interfaz.
- Saltos:
    - Origen: Tarea o acción origen del salto.
    - Destino: Tarea o acción destino del salto.

4.6.1.e Usuario

Con este tipo de documentos podremos gestionar los usuarios del sistema y sus atributos, roles y configuraciones.

Los datos de este documento son:

- Tercero: Tercero asociado al usuario.
- Alias: Alias del usuario para mostrar en la interfaz.
- Tipo de autenticación: Sistema al que se consultará para validar el usuario (Base de datos, LDAP, Active Directory, Single Sing On, etc).
- Nombre de usuario.
- Contraseña.
- Email.

4.6.1.f Rol

Con este tipo de documentos podremos gestionar los roles del sistema que luego

podrán ser usados por otros elementos del sistema para conceder o denegar accesos.

Los datos de este documento son:

- Título: Nombre del Rol.
- Referencia: Cadena de caracteres para identificar el rol.
- Descripción del rol.

4.6.1.g Terceros

Con este tipo de documentos podremos gestionar los terceros.

Los datos de este documento son:

- Tipo: Empresa/Persona/Agente.
- Nombre: Nombre completo del tercero.
- CIF: Identificación fiscal del tercero (si procede).

4.6.1.h Documento genérico

Es un documento sin ningún destino prefijado, destinado a gestionar todos aquellos datos para los que no es necesario diseñar un documento específico o para ser usado como plantilla para el futuro diseño de un documento. Permite editar todos los atributos básicos de un documento y como metadatos contiene una cadena de caracteres que el usuario podrá rellenar en cualquier formato.

4.6.2 Documentos propios de la empresa

Para hacer totalmente funcional la aplicación para una empresa será imprescindible diseñar más documentos de los propios del sistema. Estos documentos habrá que diseñarlo en común con nuestra empresa cliente para adaptarlo a sus gustos y necesidades. No obstante, con el fin de adelantarnos a las circunstancias y comprobar la funcionalidad del sistema, como parte del proyecto se han diseñado una serie de documentos (a modo de plantilla) que casi con seguridad serán imprescindibles para cualquier empresa

4.6.2.a Factura

Permite almacenar los datos típicos de una factura, esto es:

- Fecha de facturación.
- Número de factura.

- Código de cliente.
- CIF del cliente.
- Fecha de vencimiento.
- Dirección de facturación.
- Conceptos valorados.
- Base imponible, I.V.A. y total.
- Estado del pago.

4.6.2.b Albarán

Permite almacenar los datos típicos de un albarán, esto es:

- Fecha de albarán.
- Número de albarán.
- Cliente.
- Dirección de envío.
- Conceptos.

4.7 Tecnologías usadas

JSON

Acrónimo de "*JavaScript Object Notation".* Es un formato ligero de intercambio de información que permite estructurar los datos de forma que sea fácilmente legible por cualquier sistema.

HTML5

Acrónimo de "*HyperText Markup Language*, versión 5". Es un lenguaje diseñado para la creación de páginas web. A base de una serie de etiquetas le indica al navegador web qué información se desea ofrecer al usuario y como se debe organizar y presentar dicha información. La versión 5 añade nuevas etiquetas que permiten enriquecer la semántica de la página web y adaptarse mejor a los nuevos tipos de dispositivos y a las nuevas tecnologías existentes.

PHP

Acrónimo recursivo de *PHP: Hypertext Preprocessor.* Es un lenguaje de programación diseñado específicamente para la creación de páginas HTML dinámicas. A base de las estructuras de control de flujo y el resto de operaciones comunes de cualquier otro lenguaje de programación, PHP permite generar contenidos en el formato deseado por el desarrollador (generalmente HTML).

JavaScript

Es un lenguaje de programación que al igual que PHP está especialmente diseñado para la creación de páginas web dinámicas. Este código, a diferencia de PHP, se ejecuta en el lado del cliente (es decir lo interpreta el propio navegador web) y permite modificar cualquier elemento de la web devuelta por el servidor.

jQuery

Es una librería de código abierto escrita en javascript diseñada para interactuar fácilmente con los elementos y etiquetas de las web HTML. Esta librería no sólo agiliza la programación en JavaScript sino que permite tener un código más ordenado, legible y limpio.

PUSH

Es una tecnología que permite a los clientes recibir notificaciones directamente de los servidores. Para nuestra interfaz web hemos utilizado el objeto “EventSource” de JavaScript que nos permitirá que el servidor notifique y actualice nuestra interfaz web si cualquier otro usuario ha realizado algún cambio

MySQL

Es un sistema gestor de base de datos relacional utilizado muy comúnmente para el desarrollo de aplicaciones web por su potente y sencilla integración con PHP.

APC

Acrónimo de “*Alternative PHP Cache”.* Es un módulo diseñado para PHP que permite almacenar en cache el código compilado de nuestros scripts PHP. Los scripts PHP deben ser accesibles por el servidor web sin compilar y es por eso que cada vez que necesiten ejecutarse deberán compilarse previamente, esto supone un exceso de carga de trabajo innecesario por lo que en entornos de producción donde los scripts no son modificados es aconsejable habilitar dicho módulo.

5 Listas y tablas

5.1 Glosario de términos

5.1.1 Vocabulario del modelo conceptual

- Tercero: Llamaremos tercero a cualquier persona o empresa con la que nuestro cliente tendrá una relación comercial. Podrán ser clientes, proveedores, empleados, bancos, etc.

- Documento: Será una entidad que usaremos para todos los datos necesarios para la labor comercial de nuestro cliente. Siempre que necesitemos guardar algún tipo de información en el sistema lo haremos haciendo referencia a esta entidad. El formato del documento será abierto pudiendo contener cualquier tipo de información (JSON, XML, PDFs, hojas de cálculo, fotografías, enlaces a páginas webs, etc). Ejemplos:

    - Para las entradas de productos en almacén podremos asociar un documento que será el albarán (Un documento escaneado en PDF).

    - Las facturas generadas y recibidas (un fichero XML que contenga la información del documento).

    - Cuando una tarea consista en generar un informe, este será el documento de salida (Un fichero de MS Excel).

    - Cuando abramos una tarea de tipo “ruta de reparto”, el documento de entrada de dicha tarea será el itinerario a seguir (Un fichero KMZ).

    - Nuestros productos en venta podrán tener un catálogo asociado (ficheros PDF).

    - Una tareas de “solicitud de repuesto” podrá tener un documento asociado que sea el enlace a una tienda online donde se suministra.

- **Plan General de Contabilidad:** (PGC) es el texto legal que regula la contabilidad de las empresas en España.

- Principio de no compensación:Según la normativa contable no se permite compensar cuentas de activo con cuentas de pasivo, ni gastos e ingresos aunque compartan el mismo origen.

- Modelo 111 de la AT: Retenciones e Ingresos a Cuenta del IRPF. Rendimientos del trabajo y de actividades económicas, premios y determinadas

ganancias patrimoniales e imputaciones de Renta.

- **Modelo 115 de la AT:** Retenciones e ingresos a cuenta. Rentas o rendimientos procedentes del arrendamiento o subarrendamiento de inmuebles urbanos.
- **Modelo 303 de la AT**: Autoliquidación del I.V.A.
- **Modelo 347 de la AT**: Declaración anual de operaciones con terceras personas.
- **Norma 19 de la AEB:** Adeudos Directos SEPA en fichero electrónico.

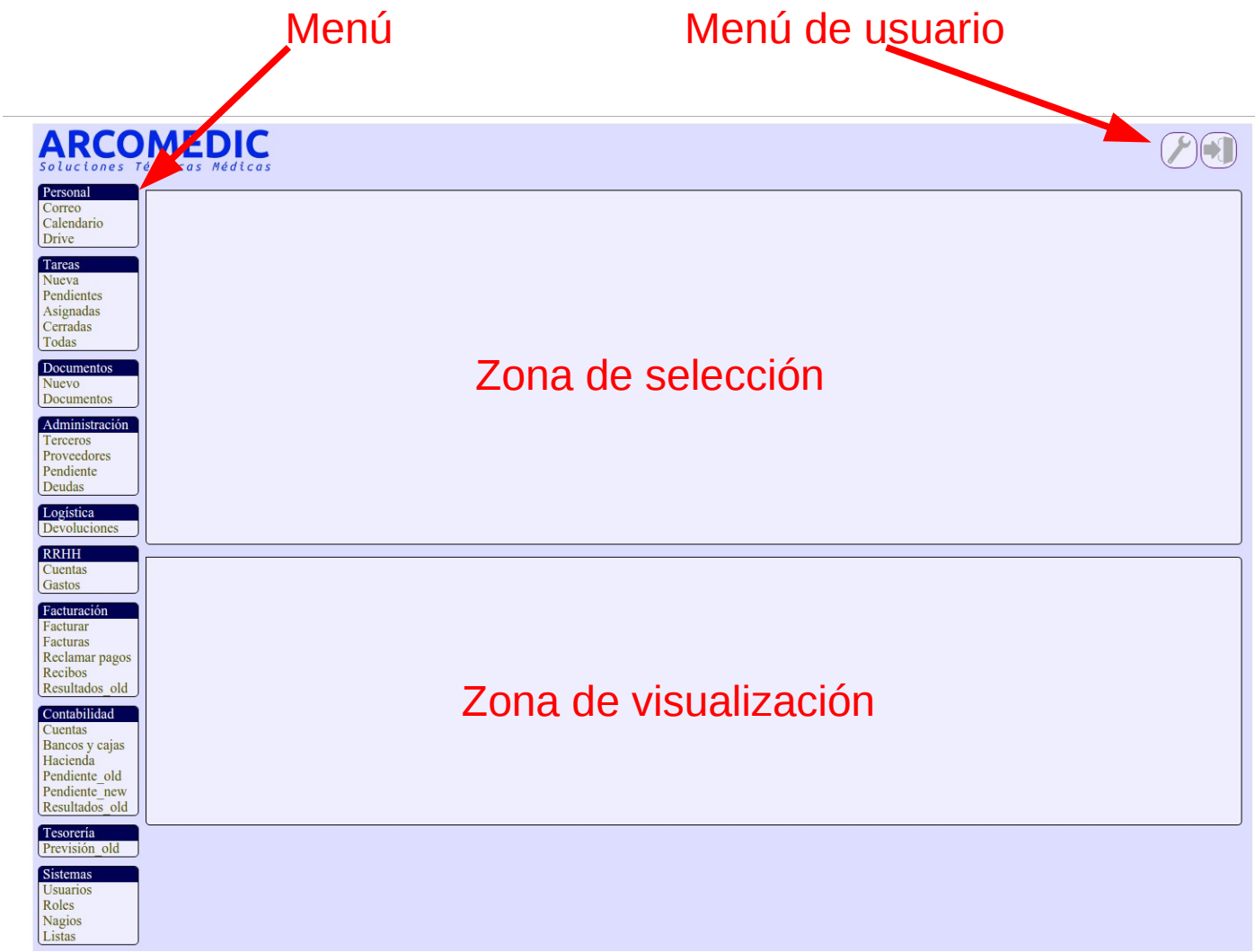
5.1.2 Vocabulario técnico

- **Núcleo de las herramientas**: Será el componente principal de nuestro software, no debe incluir ninguna referencia a software específico creado para alguno de nuestros clientes.
- **Extensiones de herramientas**: Serán ampliaciones software muy específicas para nuestro cliente. Debe evitarse su uso intentando para un requisito nuevo de algún cliente crear una solución genérica que podamos incluir en el núcleo.

6 Anexos

6.1 Manual de usuario

6.1.1 Presentación de la interfaz



- Menú: Presenta en pantalla enlaces a diferentes apartados de la aplicación
- Menú de usuario: Acceso a las principales opciones de usuario.
- Zona de selección: Normalmente destinado a mostrar listados desde donde seleccionar los elementos a los que queremos acceder.
- Zona de visualización: Zona dónde se mostrarán los datos seleccionados.

6.1.1.a Menús

- Enlace: Son vínculos a los diferentes apartados de la aplicación. Estos pueden ser enlaces a:
    - Submenús.
    - Listados.
    - Documentos.
    - Enlaces externos (URLs a otros servicios web)
- Bloques: Los enlaces se pueden agrupar en bloques cuando tengan relación entre si.

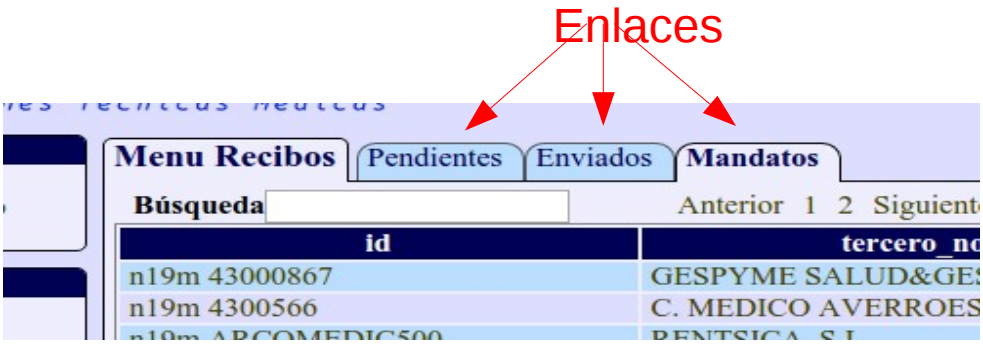
6.1.1.b Menú de usuario

- Configuración de usuario: Acceso a las diferentes opciones del usuario actualmente logueado.
- Cerrar sesión: Terminar la sesión del usuario activo.

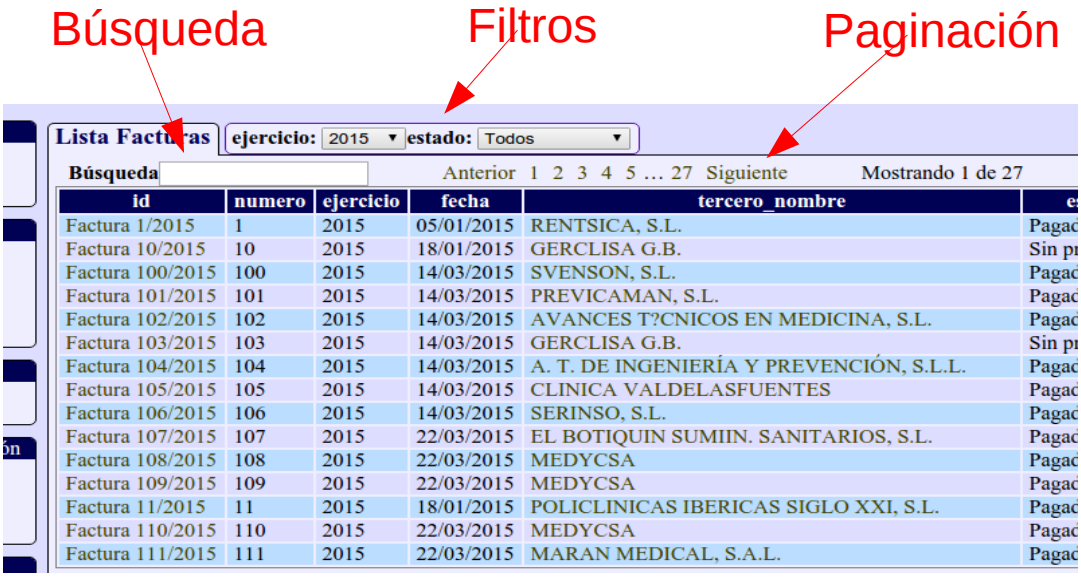
6.1.2 Elementos

6.1.2.a Zona de selección

- **Submenús:** Son menús de segundo nivel, se muestran en la zona de selección como pestañas.



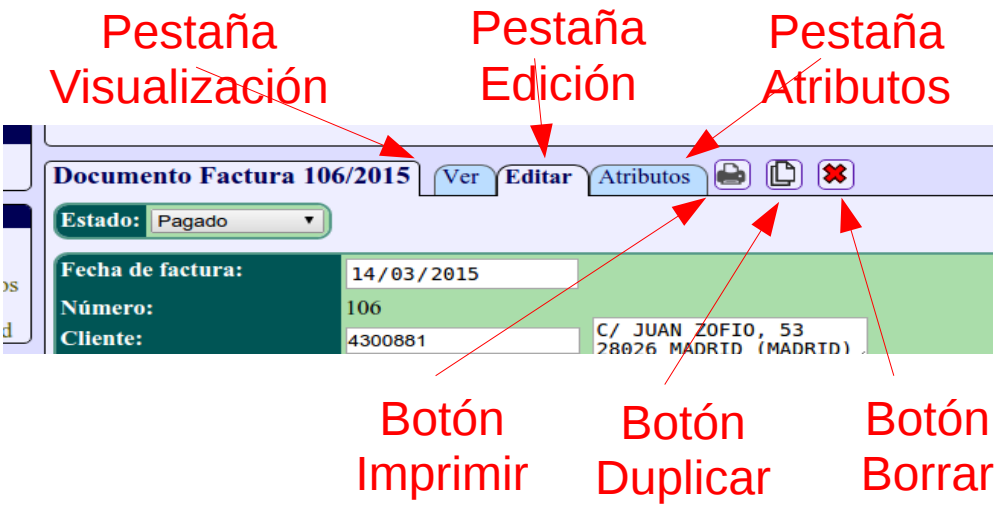
- **Listas:** Listados que muestras información y dan acceso a la visualización de elementos.



  - Búsqueda: Cuadro de texto que permite buscar elementos dentro del listado.
  - Filtros: Campos seleccionables que permiten ajustar los elementos que se mostrarán en el listado.
  - Paginación: Menú de navegación cuando el listado ocupa más de una página.

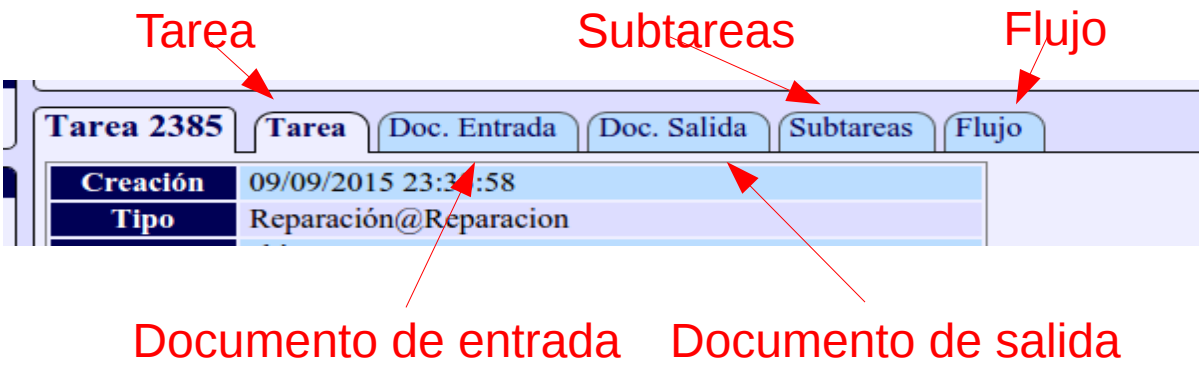
6.1.2.b Zona de visualización

- Documento:



  - Pestaña Visualización: Muestra el documento para su lectura.
  - Pestaña Edición: Muestra el documento para ser editado.
  - Pestaña Atributos: Acceso a los atributos y permisos del documento.

- Tarea:

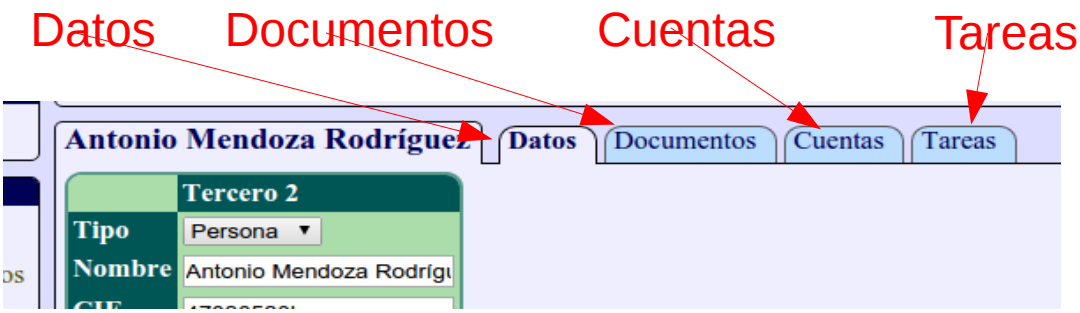


  - Tarea: Acceso a la información básica de la tarea.
  - Documento de entrada: Muestra el documento de entrada de la tarea (en modo visualización).
  - Documento de salida: Muestra el documento de salida de la tarea (en modo edición si la tarea está asignada al usuario

logueado y en modo visualización en caso contrario).

- Subtareas: Acceso al listado de todas las “tareas hija” de dicha tarea.
- Flujo: Acceso a la información del flujo al que pertenece la tarea.

- Tercero:

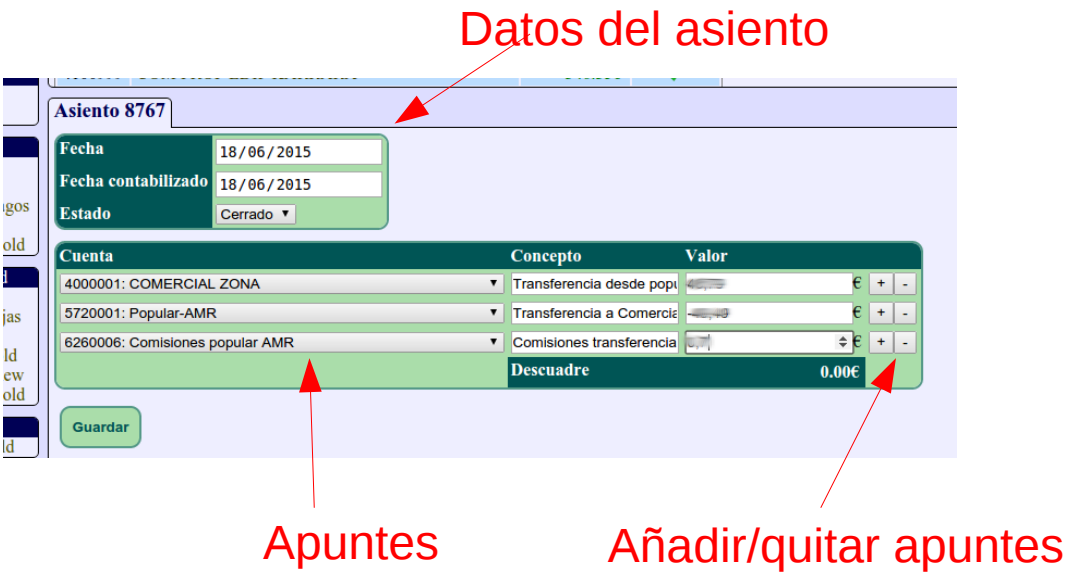


- Datos: Muestra el documento de Tipo “Tercero” correspondiente.
- Documentos: Muestra un listado de todos los documentos asociados al tercero.
- Cuentas: Muestra un listado de todas las cuentas contables asociadas al tercero.
- Tareas: Muestra un listado de todas las tareas asignadas al tercero.

- Cuenta:

Ejercicio: Permite seleccionar el año que se quiere visualizar.

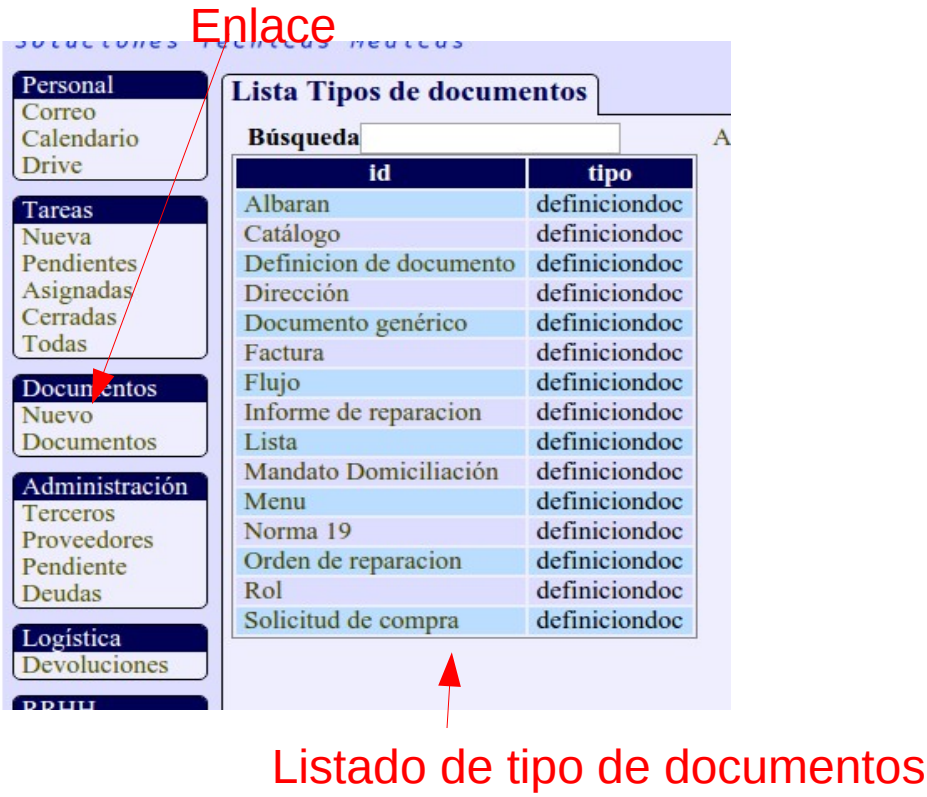
- Asiento:



6.1.3 Gestión de documentos

6.1.3.a Creación de documentos

Para crear un nuevo documento acceda en primer lugar al listado de tipos de documentos. Por defecto a este listado se puede acceder desde el enlace “Nuevo” del bloque “Documentos” del menú.



6.1.3.b Operaciones con documentos

- Ver: Seleccionando esta pestaña se mostrará el documento para su lectura.
- Editar: Seleccionando esta pestaña se mostrará el formulario para editar dicho documento.
- Atributos: Desde esta pestaña se pueden editar los atributos y permisos del documento.

- Imprimir Documento : Muestra el cuadro de dialogo para imprimir el documento.
- Duplicar documento : Crea un nuevo documento con los mismos datos.

- Eliminar documento : Elimina el documento.

6.1.4 Gestión de tareas

- Nueva: Muestra el listado de flujos disponible.
- Pendientes: Muestra todas las tareas no asignadas.
- Asignadas: Muestra el listado de tareas abiertas asignadas al usuario actual.
- Cerradas: Muestra el listado de tareas cerradas asignadas al usuario actual.
- Todas: Muestra el listado de todas las tareas asignadas al usuario actual.

6.1.4.a Iniciar nuevo flujo

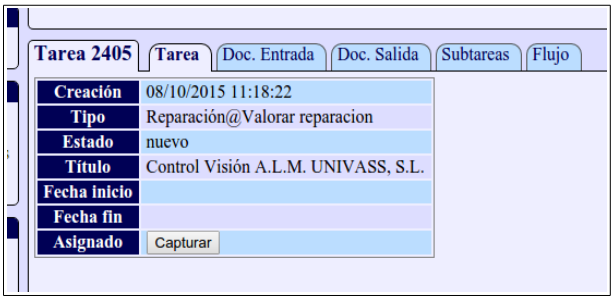
Seleccione un flujo del listado de flujos. En primer lugar debe seleccionar los datos del flujo y el tipo de tarea de inicio (si hubiese más de uno).

Una vez seleccionado el tipo de tarea de inicio se mostrará el formulario correspondiente al documento de entrada de la tarea.

Para finalizar pulse el botón “Crear”. La Tarea se generará y estará disponible en la lista de “Tareas pendientes”.

6.1.4.b Asignarse una tarea

Acceda al listado de “Tareas pendientes”. Seleccione la tarea que desee explorar.



Antes de asignarse la tarea puede explorar entre las pestañas “Tarea”, “Doc. Entrada”, “Subtareas” y “Flujo” para obtener más información acerca de la tarea seleccionado. En caso de querer asignarse la tarea pulse el botón “Capturar” de la pestaña “Tarea”, en ese momento la tarea pasará a estar asignada al usuario logueado y estará disponible en su lista de “Tareas asignadas”.

6.1.4.c Completar una tarea

En primer lugar, abra una tarea asignada a usted. Seleccione la pestaña “Doc. Salida”.

Rellene el formulario del documento y pulse el botón “Guardar” una vez cumplimentado. A continuación seleccione la pestaña “Tarea”.

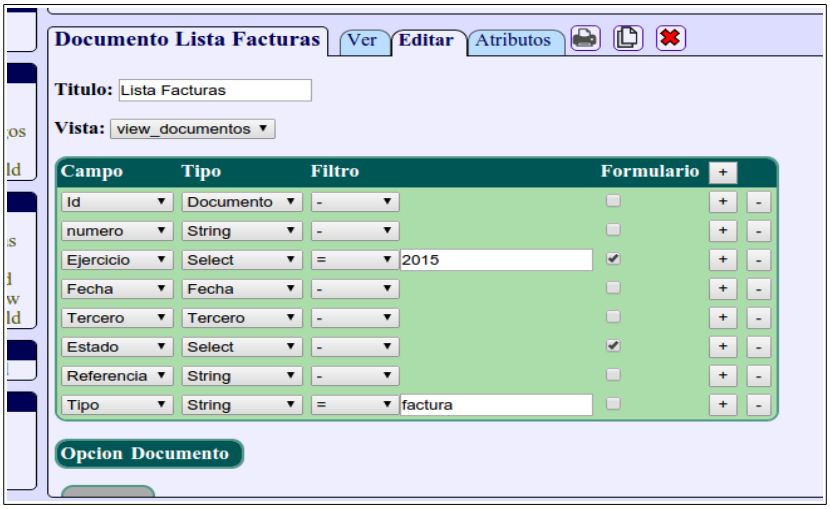
Seleccione alguna de las opciones de avance configuradas en el flujo. Dependiendo del tipo de avance se crearan las tareas correspondientes (Ver diseño de flujos).

En caso de que la tarea sea la última del flujo únicamente se mostrará la opción de avance “Cerrar”, en este caso se cerrará la tarea y el flujo se dará por terminado.

6.1.5 Configuración de la interfaz

6.1.5.a Creación de listas.

Para crear una nueva lista cree un nuevo documento de tipo “Lista”.



Titulo: Nombre que se le asignará a la lista.

Vista: Origen de los datos que mostraremos en la lista. Las opciones son “Tareas, “Documentos”, “Cuentas”, “Diario”, “Asientos”, “Otro”.

Campo: Columna que se mostrará en el listado.

Tipo: Formato en el que se mostrará el campo. Dependiendo del tipo seleccionado el campo se mostrará de una forma u otra. Por ejemplo, un campo “ID” de la vista “documentos” se mostrará como un número si se selecciona el tipo “String” y como un enlace al documento si se selecciona el tipo “Documento”.

Filtro: Condiciones que debe cumplir el campo para que sea mostrado en la lista.

Formulario: Marcando esta columna se mostrará el campo en el formulario de filtros del listado. En caso de haber fijado un filtro para este campo este indicará la opción por defecto en el formulario de filtros.

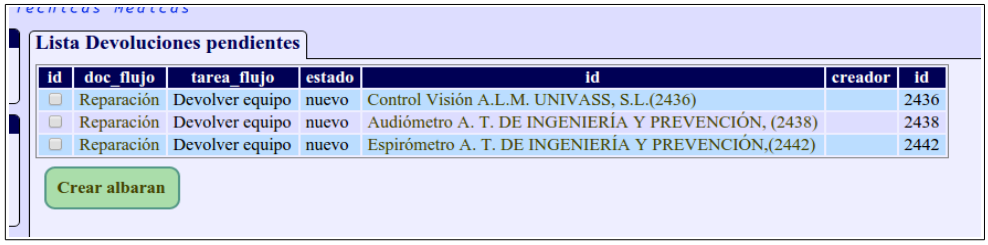
id	numero	ejercicio	fecha	tercero_nombre	estado	referencia	tipo
Factura 1/2015	1	2015	05/01/2015	RENTSICA, S.L.	Pagado		factura
Factura 10/2015	10	2015	18/01/2015	GERCLISA G.B.	Sin previsión		factura
Factura 100/2015	100	2015	14/03/2015	SVENSON, S.L.	Pagado		factura
Factura 101/2015	101	2015	14/03/2015	PREVICAMAN, S.L.	Pagado		factura
Factura 102/2015	102	2015	14/03/2015	AVANCES T?CNICOS EN MEDICINA, S.L.	Pagado		factura
Factura 103/2015	103	2015	14/03/2015	GERCLISA G.B.	Sin previsión		factura
Factura 104/2015	104	2015	14/03/2015	A. T. DE INGENIERÍA Y PREVENCIÓN, S.L.L.	Pagado		factura
Factura 105/2015	105	2015	14/03/2015	CLINICA VALDELASFUENTES	Pagado		factura
Factura 106/2015	106	2015	14/03/2015	SERINSO, S.L.	Pagado		factura
Factura 107/2015	107	2015	22/03/2015	EL BOTIQUIN SUMIIN. SANITARIOS, S.L.	Pagado		factura
Factura 108/2015	108	2015	22/03/2015	MEDYCSA	Pagado		factura
Factura 109/2015	109	2015	22/03/2015	MEDYCSA	Pagado		factura
Factura 11/2015	11	2015	18/01/2015	POLICLINICAS IBERICAS SIGLO XXI, S.L.	Pagado		factura
Factura 110/2015	110	2015	22/03/2015	MEDYCSA	Pagado		factura
Factura 111/2015	111	2015	22/03/2015	MARAN MEDICAL, S.A.L.	Pagado		factura

Resultado final del listado del ejemplo.

Antes de dar por finalizada la configuración de la lista puede pulsar la pestaña “Ver” para previsualizar un ejemplo del resultado final”.

Listas seleccionables

Las listas seleccionables nos permiten mostrar un listado con una columna de selección. Al pie de la lista podremos añadir tipos de documentos que podremos crear a partir de los registros seleccionados.

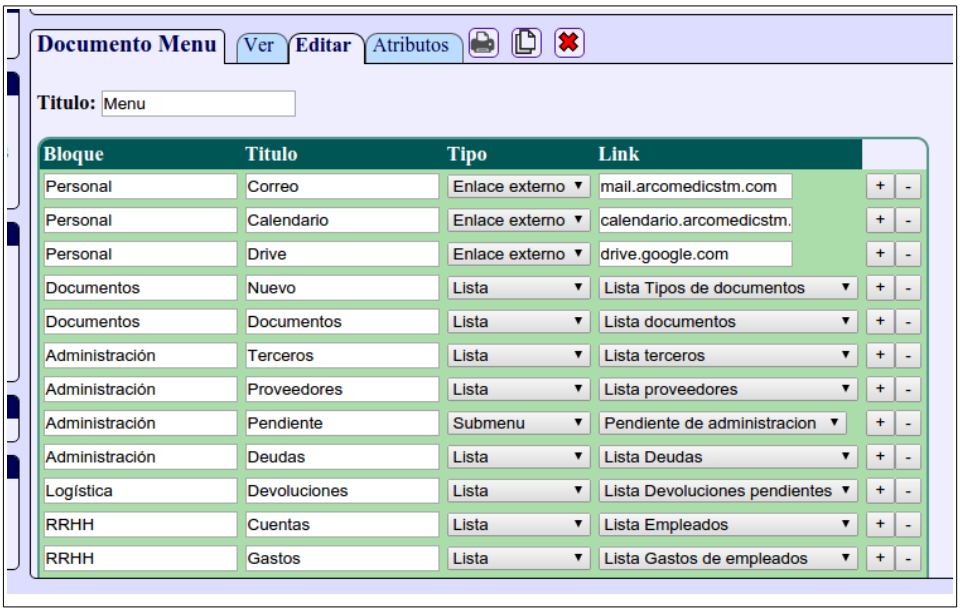


Para crear una lista seleccionable primero debemos añadir una columna con el campo que usaremos para identificar el registro (normalmente el “ID”) y seleccionar dicho campo como tipo “Check”.

A continuación, añada en el listado de opciones tantas opciones como sean necesarias indicando el texto que se mostrará y el tipo de documento que se creará.

6.1.5.b Creación de menús

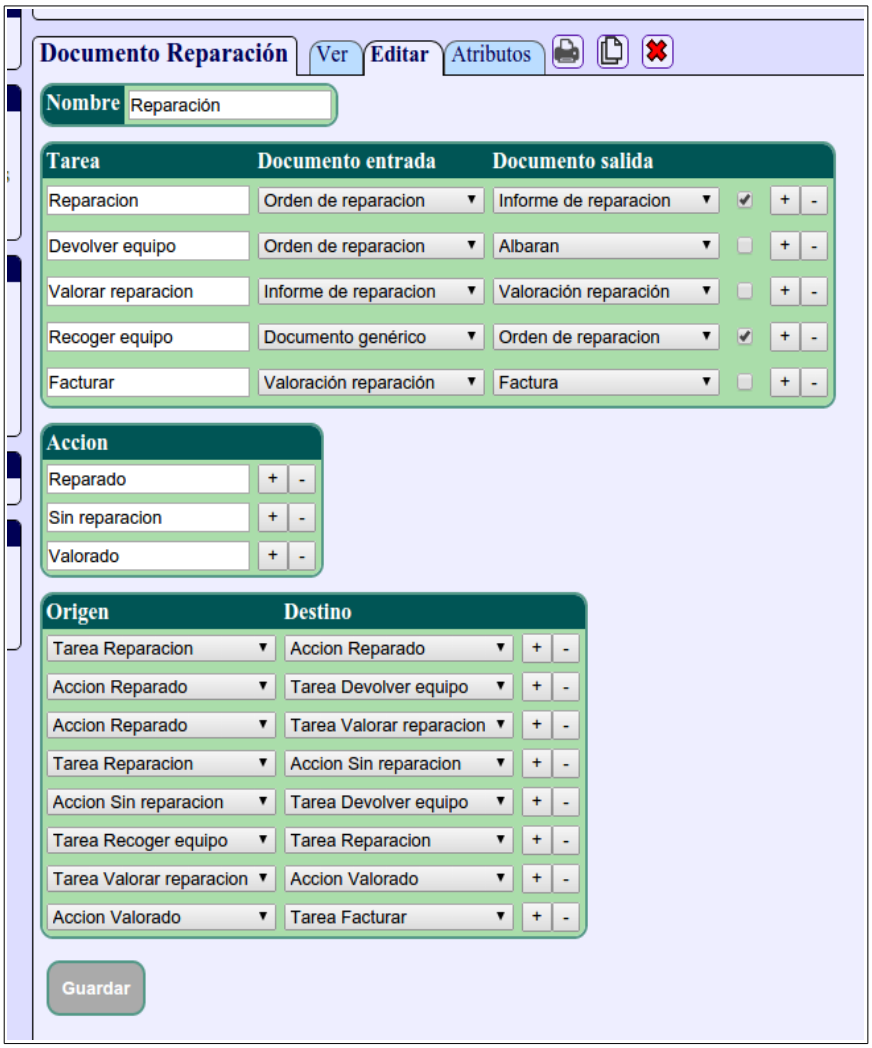
Tanto los menús como los submenús se crean con el tipo de documento “Menú”.



- **Bloque**: Título del bloque que contendrá el enlace, los enlaces consecutivos con el mismo bloque irán agrupados.
- **Título**: Texto a mostrar en el enlace.
- **Tipo**: Tipo de enlace:
    - Enlace externo: Enlace a una URL de un servicio web externo.
    - Lista: Enlace a una lista.
    - Submenú: Enlace a un submenú.
- **Link**: Elemento al que hace referencia el enlace.

6.1.5.c Configuración de flujos

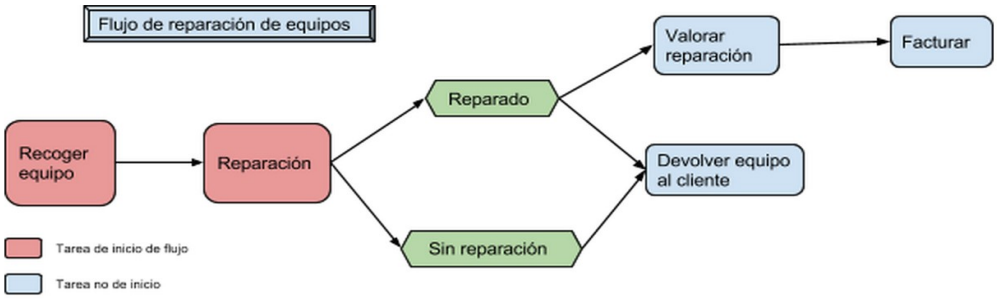
Seleccione crear un nuevo documento de tipo “Flujo”.



- Nombre: Título del flujo.
- Tareas:
    - Tarea: Título de la tarea.
    - Documento de entrada: Tipo de documento de entrada.
    - Documento de salida: Tipo de documento de salida.
    - Inicio: Marcando esta casilla se creará la tarea como “inicial”.
- Acciones:
    - Acción: Nombre de la acción.
- Saltos:
    - Origen: Tarea o acción de origen.

- Destino: Tarea o acción de destino.

Para diseñar los flujos es preferible pensar un diagrama como el del ejemplo:

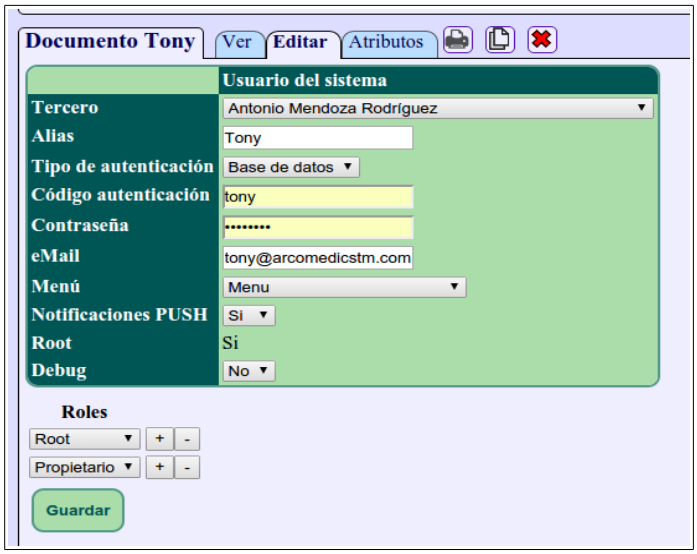


A la hora de rellenar el formulario habrá que convertir los rectángulos en tareas, los hexágonos en acciones y las aristas en saltos.

6.1.6 Gestión de usuarios y roles.

6.1.6.a Creación y gestión de usuarios

Los usuarios del sistema son tratados como documentos, por tanto crear y modificar usuarios será como crear y modificar documentos del tipos “Usuario”.



- **Tercero**: Tercero asociado al usuario.
- **Alias**: Nombre que se mostrara en la interfaz.
- **Tipo de autenticación**: Sistema que se consultará para verificar al usuario. En

principio únicamente se puede validar al usuario por base de datos pero el sistema está preparado para añadir otros sistemas de autenticación como LDAP, Active Directory, Single Sign On, etc.

- **Contraseña**: Contraseña del usuario.
- **Email**: Dirección de correo electrónico del usuario, útil para recuperar contraseñas.
- **Menú**: Menú que se mostrará al usuario.
- **Notificaciones PUSH**: Activar las notificaciones PUSH.
- **Root**: Concesión de privilegios de administrador.
- **Debug**: Marcando el modo debug se mostrará información adicional en la interfaz web del usuario.

6.1.6.b Creación y gestión de roles

Los roles del sistema son tratados como documentos, por tanto crear y modificar usuarios será como crear y modificar documentos del tipos “Rol”.

- **Título**: Texto a mostrar como nombre del rol.
- **Referencia**: Código para identificar el rol.
- **Descripción**: Breve descripción del rol

6.2 Guía de diseño y programación de documentos

A la hora de crear un nuevo tipo de documento, el desarrollador deberá diseñar la lógica del mismo y trasladar dicha lógica a los ficheros que forman parte del documento. Esta lógica se reparte en 3 ficheros:

- index.php: El SCRIPT principal del documento y al que el sistema llamará cada vez que este sea abierto.
- save.php: SCRIPT de guardado del documento.

- delete.php: SCRIPT de borrado del documento.

6.2.1 Index.php

Este es el fichero principal del documento y será llamado por el sistema cada vez que este sea abierto. El SCRIPT será invocado con una serie de variables de entorno que se deberán interpretar a fin de mostrar la información del modo adecuado:

- $documento: Será un array con todos los datos del documento. Si esta variable no está definida deberá interpretarse que se trata de un documento nuevo y vacío.
- $modo_apertura_documento: Podrá ser “ver” / “edicion” / “imprimir” para indicar el objetivo con el que se ha abierto el documento.
- $_REQUEST['ids']: Lista de los campos seleccionados en el “listado seleccionable” desde el que se abrió el documento.

Se recomienda dividir el SCRIPT index.php en bloques:

6.2.1.a Apertura del documento:

En esta sección se recuperarán los datos del documento que deseamos abrir o se inicializaran los valores si se trata de un nuevo documento.

Ejemplo:

```
if (isset($documento)){                    //Comprobamos si estamos abriendo un documento existente.
        $valor_numero=$documento['numero'];
        $valor_referencia=$documento['referencia'];
        $json=myjson_decode($documento['meta']);                //Parseamos los metadatos en formato JSON
        $valor_datos=$json->form_documento_datos;
}else{  //Vamos a crear un documento nuevo
        if (isset($_REQUEST['ids'])){              //Creamos un nuevo documento a partir de los registros seleccionados en un listado.
                $valor_numero='';
                foreach ($_REQUEST['ids'] as $item) {
                        $item_doc=SQLquery_documento($item);
                        $valor_referencia=$valor_referencia . “, “ $item_doc['referencia'];
                        $valor_datos=$valor_datos . “, “ $item_doc['meta'];
                }
        }else{  //Establecemos los valores por defecto para el nuevo documento.
                $valor_numero='';
                $valor_referencia='';
```

```
                $valor_datos='';
        }
```

6.2.1.b Preparación de los campos

En esta sección se preparan los campos para mostrarlos. En caso de abrir un documento para su edición los campos se transformarán en elementos de formularios:

Ejemplo:

```
if ($modo_apertura_documento == 'edicion'){
        $campo_numero=printString($valor_numero);
        $campo_referencia='<input name="form_documento_referencia" value="' . $valor_referencia . '"/>';
        $campo_datos='<textarea name="form_documento_datos">' . $valor_datos . '</textarea>';
}else{
        $campo_numero=printString($valor_numero);
        $campo_referencia=printString($valor_referencia);
        $campo_datos=printString($valor_datos);
}
```

6.2.1.c Presentación de los campos

Una vez preparados los campos se deben mostrar de forma ordenada en formato HTML:

Ejemplo:

```
<table class='formulario'><tbody>
        <tr>
                <td></td>
                <th>Documento genérico (<?php echo $campo_numero ?>)</th>
        </tr>
        <tr>
                <th>Referencia</th>
                <td><?php echo $campo_referencia ?></td>
        </tr>
        <tr>
                <th>Datos</th>
                <td><?php echo $campo_datos ?></td>
        </tr>
</tbody></table>
```

6.2.2 Save.php

Este SCRIP será invocado cada vez que el usuario presione el botón “Guardar” al editar un documento. Su objetivo es recuperar la información del formulario y prepararlos para ser guardados en la base de datos.

Ejemplo:

```
if(isset($_REQUEST['form_doc_id'])){                //Si ya tiene asignado un ID estamos guardando cambios en un documento.
        $id=$_REQUEST['form_doc_id'];
}else{                // Si no está definido el ID estamos creando un nuevo documento.
        $id=null;
}

// Preparamos los datos a guardar.
$tipo='generico';
$titulo='Solicitud genérica';
$referencia=$_REQUEST['form_documento_referencia'];
$meta=json_encode($_REQUEST);
$tercero=null;

$doc=SQLsave_doc($id,$tipo,$titulo,$referencia,$meta,$tercero);      // Llamamos a la función para guardar el documento.
SQLsave_doc_permiso($doc,null,'ver');              // Asiganmos los permisos al documento.
SQLsave_doc_permiso($doc,SQLquery_documentos('rol',null,null,'propietario'),'editar');
echo "$doc";                    // Mostramos por la salida estándar el nuevo ID del documento (Podrá ser utilizado por otras funciones
del sistema)
```

6.2.3 Delete.php

```
if(isset($_REQUEST['form_doc_id'])){                //Comprobamos que se ha definido el ID del documento a borrar.
        SQLdelete_doc($_REQUEST['form_doc_id']);      // Llamamos a la función para borrar el documento.
}
```

6.2.4 action_functions.inc

Este SCRIPT contiene funciones que facilitarán el diseño de documentos. Por ejemplo:

- **permiso_usuario($docid,$permiso):** Comprueba si el usuario actual tiene un permiso concreto sobre un documento.
- **mksqltime($fecha):** Convierte una fecha en formato SQL.

- **printString($str):** Muestra un String en la interfaz.
- **printBool($bool):** Muestra un valor de tipo booleano en la interfaz.
- **printValor($valor):** Muestra un importe en la interfaz.
- **printValorCols($valor):** Muestra un importe en la interfaz en formato de dos columnas (DEBE/HABER).
- **printFecha($fecha):** Muestra una fecha en la interfaz.
- **printFechaHora($fecha):** Muestra una fecha y una hora en la interfaz.
- **printDocumento($id,$nombre=null):** Muestra un enlace a un documento, si no se especifica el nombre se busca el título del documento en la base de datos.
- **printTercero($id,$nombre=null):** Muestra un enlace a un tercero, si no se especifica el nombre se busca el nombre del tercero en la base de datos.
- **printTarea($id,$nombre=null):** Muestra un enlace a una tarea, si no se especifica el nombre se busca el título de la tarea en la base de datos.
- **printCuenta($id,$nombre=null):** Muestra un enlace a una cuenta, si no se especifica el nombre, se busca el nombre de la cuenta en la base de datos.
- **printAsiento($asiento):** Muestra un enlace a un asiento.
- **printLista($sql,$campos,$tipos,$tableid=''):** Crea una tabla para mostrar los resultados de una query a base de datos.
- **form2where($forms_campos,$forms_tipos,$forms_values):** Convierte un formulario en una claúsula WHERE de SQL.
- **doc2where($campos,$comps,$compvalues):** Convierte los filtros de un documento de tipo “lista” en una claúsula WHERE de SQL.

6.2.5 action_db.inc

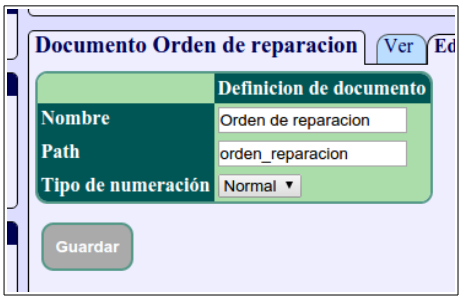
Contiene todas las funciones que permiten interactuar con la base de datos. Por ejemplo:

- **SQLquery($sql):** Realiza una consulta a la base de datos y devuelve la lista de registros obtenidos.
- **SQLsave($sql):** Ejecuta una query a la base de datos.
- **SQLqueryrow($sql):** Realiza una consulta a la base de datos que se presupone devolverá un único registro, devuelve directamente el registro devuelto.

- **SQLquery_documento($id):** Devuelve un documento a partir de su ID.
- **SQLquery_tercero($tercero):** Devuelve un tercero a partir de su ID.
- **SQLquery_tarea($id):** Devuelve una tarea a partir de su ID.
- **SQLquery_asiento($id):** Devuelve un asiento a partir de su ID.
- **SQLquery_cuenta($id):** Devuelve una cuenta a partir de su ID.
- **SQLquery_documentos($tipo=null,$numero,$tercero=null,$referencia):** Devuelve un listado de documentos del tipo, número, referencia y tercero indicado.
- **SQLquery_tareas($tipo,$estado,$flujo):** Devuelve el listado de tareas del tipo, estado y flujo indicado.
- **SQLquery_tareas_hija($padre):** Devuelve el listado de tareas hija de otra tarea.
- **SQLquery_tareas_doc($id,$tipo=null):** Devuelve las tareas asociadas al documento indicado. El tipo indica si la asociación será como documento de entrada, salida o ambos.
- **SQLsave_doc($id,$tipo,$titulo,$referencia=null,$meta=null,$tercero=null, $activo=null,$fecha='NOW()',$estado=''):** Guarda un documento.
- **SQLsave_doc_permiso($docid,$rol,$permiso):** Guarda los permisos de un documento.
- **SQLsave_doc_permisodel($docid):** Borra los permisos de un documento.
- **SQLsave_tarea($id,$padre,$doc_flujo,$tarea_flujo,$titulo,$urgencia, $vencimiento):** Guarda una tarea.
- **SQLsave_asignar_doc($tarea,$tipo,$documento):** Asigna un documento a una tarea. El tipo indica si se asignará como documento de entrada o salida.
- **SQLsave_capturar_tarea($id):** Captura una tarea y la asigna al usuario actual.
- **SQLsave_tarea_cerrar($id):** Cierra una tarea.
- **SQLsave_asiento($id,$fecha,$contabilizado):** Guarda un asiento.
- **SQLsave_apunte($asiento,$id,$concepto,$valor):** Guarda un apunte.
- **SQLsave_apuntesdel($id):** Borra todos los apuntes de un asiento.

6.2.6 Alta del nuevo tipo de documentos

Una vez creados los fichero los ficheros necesarios para el funcionamiento del documento habrá que darlo de alta en la aplicación. Para ello crearemos un nuevo documento del tipo “Definición de documento”.



En dicho documentos deberemos indicar:

- **Nombre:** Título del tipo de documento.
- **Path**: Directorio donde hemos guardado los ficheros del documento.
- T**ipo de numeración**: Forma en la que se numeraran los documentos. Puede ser:
    - **Normal**: Los número de documento se asignaran consecutivos.
    - **Anual**: Los número de documentos se reiniciarán cada año (útil para facturas).